Laura Ohrndorf

Entwicklung und Validierung eines Instruments zur Messung des Wissens über Fehlvorstellungen in der Informatik

Dissertation
zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)

Universität Paderborn

Fachbereich Elektrotechnik, Informatik und Mathematik der Universität Paderborn

Paderborn, Mai 2016

Gutachter:
Prof. Dr. Johannes Magenheim
Prof. Dr. Carsten Schulte

Datum der mündlichen Prüfung:
19. Oktober 2016

Kurzfassung

Fehlvorstellungen sind seit mehreren Jahrzehnten ein vielbeachtetes Thema in den naturwissenschaftlichen Fächern und der Mathematik und spielen in der Ausbildung von Lehrenden eine größer werdende Rolle. Aufgrund des jungen Alters der Informatik und der Informatikdidaktik sind die Forschungsergebnisse zu Fehlvorstellungen noch unstrukturiert und wenig erprobt.

Um eine Grundlage für die weitere Forschung zu schaffen, wird in dieser Arbeit zunächst ein Überblick über Definitionen und Eigenschaften von Fehlvorstellungen gegeben. Darauf aufbauend werden vorhandene Forschungsergebnisse aus der Informatik anhand verschiedener Kriterien analysiert. Dadurch ist es möglich, das Gebiet zu strukturieren, um zum einen den aktuellen Forschungsstand einzuschätzen und zum anderen individuelle Ergebnisse bewerten und einordnen zu können.

Anschließend wird eine empirische Erhebung durchgeführt, in der Experten Konzepte der Informatik bewerten, die mit Hilfe einer Häufigkeitsanalyse aus dem ACM/IEEE Curriculum extrahiert wurden. Die Bewertung erfolgt anhand der vier Kriterien der Fundamentalen Ideen nach Schwill, sowie durch eine Einschätzung der Häufigkeit von Fehlvorstellungen zu diesem Konzept.

Zu den im vorhergehenden Schritt identifizierten relevanten Konzepten wird ein Testinstrument erstellt, mit dem das Wissen über Fehlvorstellungen gemessen werden kann. Dies geschieht auf Basis bekannter und dokumentierter Fehlvorstellungen, die in Test-Items eingebettet werden. Ein besonderes Augenmerk wird dabei auf das Format der Items gelegt, um sicherzustellen, dass diese die Gütekriterien empirischer Forschung erfüllen. Das Instrument wird in einer Studie mit Teilnehmern aus den Gruppen Studierende Lehramt Informatik, Informatik Lehrer und sonstige Informatiker eingesetzt. Die Ergebnisse werden anschließend deskriptiv statistisch ausgewertet.

Abstract

Misconceptions have been an important topic in science subjects and mathematics during the last decades and have played an increasing role in the education of teachers. Due to the rather short history of computer science education, the research results in this field are still unstructured and incomplete. This also applies to the education of teachers, where misconceptions are often ignored.

To create a basis for further research steps, this thesis gives an overview of general definitions and characteristics of misconceptions. Afterward, research results concerning the field of computer science are analyzed based on several criteria. Through this, it is possible to structure the field to evaluate the current state of research on the one hand and to assess and classify specific findings on the other hand.

Following this, an empirical study is conducted in which experts are asked to rate the most important central concepts for introductory courses extracted from the ACM/IEEE curriculum. This is done using four criterion regarding their relevance (following the approach of the Fundamental Ideas by Schwill (1994)) and one criteria concerning the frequency of misconceptions associated with them.

Based on the results of this survey, a test instrument is developed, which can be used to measure the knowledge about misconceptions. This will is done based on documented misconceptions that are embedded into the test items. Special attention is paid on the format of the items in order to assure that all quality criteria for empirical research are observed. This instrument is used to measure test persons from three different groups: students of computer science education, computer science teachers and computer scientists in other positions. The results are analyzed and described afterward.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CI	Concept Inventories
CIT	Critical Incident Technique
COACTIV	Cognitive Activation in the Classroom
DLCI	Digital Logic Concept Inventory
GI	Gesellschaft für Informatik e.V.
KCS	Teachers' Knowledge of Content and Students
KCT	Knowledge of Content and Teaching
KSM Framework	Knowledge, Strategies and Models Framework
MCTS	Methods of Teaching Computer Science
NEA	Newman's Error Analysis
PCK	Pedagogical Content Knowledge
SIGCSE	ACM Special Interest Group in Computer Science Education
SOLO	Structure of Observed Learning Outcome
TC	Threshold Concepts

1. Einleitung

1.1. Motivation und Ziel

Auch wenn der Begriff der „Fehlvorstellung“ außerhalb von Wissenschaft und Forschung selten verwendet wird, so sind diese doch in der öffentlichen Wahrnehmung der Informatik präsent. Im Jahre 2015 bekam eine Publikation mediale Aufmerksamkeit, aus der hervorging, dass in Indonesien befragte Facebook-Nutzer nicht wissen, dass sie das Internet nutzen. Heraus kam dies durch eine Studie, in der zwar 11 % der Befragten angaben, dass sie Nutzer von Facebook sind, aber ebenso angaben, dass sie das Internet nicht nutzen (Mirani, 2015). Dies ist aufgrund verschiedener struktureller und konzeptueller Gegebenheiten im Land leicht nachvollziehbar. So sind vergünstigte Internet Tarife erhältlich, mit denen lediglich Angebote von Faceboook genutzt werden können. Die Nutzer missverstehen somit, dass sie mit einem solchen Angebot einen Teil des Internets nutzen können. Dies ist zwar ein sehr allgemeines, wenig fachwissenschaftliches Beispiel, zeigt aber wie beiläufig und unbemerkt Fehlvorstellungen entstehen können. Dieses Phänomen macht vor fachlich anspruchsvolleren Themengebieten nicht halt, die das Thema der vorliegenden Arbeit sind.

In der Psychologie existieren zahlreiche Modelle, die beschreiben, wie Fehlvorstellungen entwickelt werden, wenn neues Wissen erworben wird. Ein Eckpunkt, der hierbei stets eine Rolle spielt, ist das bereits vorhandene Wissen und die Einordnung und Verknüpfung mit dem neu Erlernten. Dies geschieht im Regelfall unbewusst, ohne eine tiefer gehende Reflexion der tatsächlichen Eignung des Vorwissens (vgl. z. B. Mandl u. Friedrich, 2006). Wie in dem oben genannten Beispiel aus Indonesien erfolgen aus Beobachtungen falsche logische Schlussfolgerungen. Solche Fehlvorstellungen können über Jahre existieren und unentdeckt bleiben (vgl. z. B. Smith u. a., 1993). Genau hier entsteht eine besondere Problematik der Fehlvorstellungen,

da sie zwangsweise auch den weiteren Wissenserwerb beeinflussen und sich so vervielfältigen können.

Nicht nur der Wissenserwerb außerhalb eines strukturierten Umfeldes ist eine Quelle für Fehlvorstellungen. Mehr und mehr ist in den letzten Jahren in den Fachdidaktiken ein Bewusstsein dafür entstanden, dass Fehlvorstellungen auch im Rahmen der Lehre, nicht selten auch im fortgeschrittenen Unterricht, entstehen können. Dieses Phänomen wird von Barke (2006, S. 21) als „hausgemachte Fehlvorstellungen“ bezeichnet. Sie unterscheiden sich von den zuvor beschriebenen Fehlvorstellungen, weil sie durch Vermittlungsfehler in der Lehre entstehen und damit vermieden werden können.

Fehlvorstellungen haben in der Informatik, ebenso wie in den Naturwissenschaften, auch eine besondere Bedeutung, da Lerngruppen meist heterogen sind und ein uneinheitliches Vorwissen vorhanden ist (vgl. Schubert u. Schwill, 2011, S. 287). Fast alle Schülerinnen und Schüler haben bereits Erfahrungen im Umgang mit dem Computer gesammelt. Diese variieren allerdings von der einfachen Nutzung, z. B. dem Umgang mit einem Web-Browser und Textverarbeitungsprogrammen bis hin zur Programmierung (vgl. z. B. Magenheim u. Schulte, 2005). Da dieses Vorwissen zumeist autodidaktisch erworben wurde, entspricht es inhaltlich sowie strukturell häufig nicht den Kompetenzen, die im Schulunterricht oder in Universitätsvorlesungen erworben werden sollen. Oft existieren Lücken, die durch vorschnelle Schlussfolgerungen überbrückt werden und so zu Fehlvorstellungen werden.

Analogien und Metaphern sind in der Informatik beliebt und werden gerne verwendet. So konnte empirisch belegt werden, dass gute Analogien, die schematisch eingesetzt werden, den Lernerfolg positiv beeinflussen (Chee, 1993). Die entscheidende Frage, die leider nicht ohne Schwierigkeiten zu beantworten ist, liegt darin, welche Analogien tatsächlich „gut“ sind. Tatsächlich existieren zahlreiche Beispiele für schlechte Metaphern, die die Entwicklung eines falschen mentalen Modells provozieren und damit Fehlvorstellungen generieren (Forišek u. Steinová, 2012).

Dies führt zu der Frage, was überhaupt über Fehlvorstellungen in der Informatik bekannt ist, in welchen Themenbereichen Fehlvorstellungen besonders häufig auftreten und wie das Wissen über Fehlvorstellungen von Lehrenden erworben werden kann. Die Beantwortung dieser Fragen kann schon in Ansätzen dazu beitragen, das Bewusstsein bei Lehrenden über Fehlvorstellungen zu stärken und damit gleichzeitig den Weg für die Verbreiterung des Wissens über Fehlvorstellungen zu ebnen. Diese Arbeit soll einen Beitrag dazu liefern.

1.2. Forschungsziele und Forschungsmethodik

Das Ziel dieser Arbeit ist zunächst die Analyse und Strukturierung des Forschungsgebiets Fehlvorstellungen in der Informatik. Die Ergebnisse werden anschließend genutzt um ein Testinstrument zu entwickeln, mit dem das Wissen über Fehlvorstellungen gemessen werden kann. Das Forschungsziel umfasst die folgenden Teilziele:

F1: Analyse und Bewertung des aktuellen Forschungsstandes zu Fehlvorstellungen in der Informatik

Hier soll zunächst ein Überblick über den aktuellen Forschungsstand zu Fehlvorstellungen in der Informatik gegeben werden. Dies ist nötig, um eine Basis zu schaffen, die einerseits das weitere Vorgehen nachvollziehbar macht und begründet. Andererseits soll diese aber auch Inhalte und Ergebnisse liefern, die im weiteren Verlauf dieses Forschungsprojekts genutzt und integriert werden können. Dies bezieht sich insbesondere darauf, dass Fehlvorstellungen oder spezifische Anwendungskontexte, in denen Fehlvorstellungen auftreten, identifiziert werden, um sie später im Messinstrument implementieren zu können.

Die Analyse schließt auch eine Bewertung der vorhandenen Forschungsergebnisse mit ein, indem insbesondere die Methodik von Erhebungen vertieft betrachtet wird und grundlegende Annahmen über Fehlvorstellungen in der Informatik verglichen werden. Letzteres dient insbesondere dazu, die Vergleichbarkeit und Vereinbarkeit von Ergebnissen sicherzustellen.

F2: Analyse und Erhebung relevanter Konzepte und der Häufigkeit von Fehlvorstellungen in der Informatik

Die zur Erreichung des vorhergehenden Forschungsziels analysierten und bewerteten Forschungsergebnisse sind thematisch weit gestreut und geben wenig Aufschluss darüber, wie häufig einzelne Fehlvorstellungen in der Praxis auftreten. Gleichzeitig scheinen einzelne in den Forschungsarbeiten betrachtete Konzepte abseits der tatsächlichen Lehrpraxis zu liegen und wenig relevant zu sein. Aus diesem Grund besteht das zweite Forschungsziel darin, Konzepte zu ermitteln, die einerseits in Einführungskursen in der Informatik relevant sind und andererseits auch häufig mit Fehlvorstellungen verbunden sind.

Da verwandte Ergebnisse bereits existieren, soll hier zunächst überprüft werden, inwieweit diese methodisch adaptiert oder Teilergebnisse übernommen werden können. Anschließend soll die Erhebung relevanter Konzepte und der Häufigkeit der

mit diesen Konzepten verbundenen Fehlvorstellungen mit Experten durchgeführt werden. Daraufhin erfolgt eine empirische Auswertung, die als Ergebnis diejenigen Konzepte ermittelt, zu denen im folgenden Forschungsziel Testitems entwickelt werden sollen.

F3: Entwicklung und Einsatz eines Testinstruments zur Messung des Wissens über Fehlvorstellungen in der Informatik

Aufbauend auf der zuvor erfolgten Analyse und Erhebung soll ein Testinstrument erstellt werden, mit dem das Wissen über Fehlvorstellungen gemessen werden kann. Zu diesem Zweck sollen die zuvor ermittelten Konzepte und damit verbundene Fehlvorstellungen in Form eines Testinstrument operationalisiert werden. Des Weiteren soll ein Fragebogen zur Erfassung relevanter demographischer Daten erstellt werden. Das Testinstrument soll mit einer repräsentativen Population, bestehend aus den drei Ausprägungen Studierende im Lehramt Informatik, Lehrer Informatik und Sonstige (im Sinne sonstiger in der Informatik tätiger Personen) erprobt werden.

Anschließend erfolgt die deskriptive statistische Auswertung der Studie. Dadurch soll zunächst ermittelt werden, ob zwischen den einzelnen gemessenen Gruppen ein signifikanter Unterschied in den Ergebnissen, sowohl bezogen auf das Testinstrument insgesamt, als auch auf einzelne Items, besteht. Dies soll unterstützt werden durch eine exemplarische Analyse von Antworten, die Aufschluss über eventuelle Perspektiven, Sichtweisen und auch Probleme bei der Bearbeitung des Testinstruments geben können.

Um die Analyse und Auswertung abzuschließen, werden die Ergebnisse zusammengefasst und das Instrument hinsichtlich seiner Eignung kritisch betrachtet. Dies bezieht sich insbesondere auf möglicherweise nötige Anpassungen, wie die Bearbeitung und Umformulierung einzelner Items. Das Forschungsziel soll durch einen Ausblick auf zukünftige Einsatzmöglichkeiten abgeschlossen werden.

1.3. Rahmenbedingungen des Forschungsprozesses und Publikationen

An dieser Stelle soll ein Überblick über die Rahmenbedingungen während des Forschungsprozesses und damit verbundene Tätigkeiten, Projekte und Publikationen gegeben werden. Um dies besser verständlich zu machen, werden einzelne

Zwischenschritte des Forschungsprozesses vorgestellt. Eine detaillierte Darstellung dieser Ergebnisse erfolgt erst im entsprechenden nachfolgenden Kapitel.

Ein Teil des in dieser Arbeit beschriebenen Forschungsprozesses entstand im Rahmen der Tätigkeit der Autorin als wissenschaftliche Mitarbeiterin an der Universität Siegen. Dies gilt für die Strukturierung und Klassifizierung von Fehlern in der Informatik, die auf dem IFIP World Congress 2013 in Torun, Polen, publiziert wurde (Ohrndorf, 2013). Im Anschluß daran erfolgte die Entwicklung von Testitems, die 2013 auf der WiPSCE Konferenz in Aarhus, Dänemark, vorgestellt und diskutiert wurden (Ohrndorf u. Schubert, 2013).

Ein Teil der zu diesem Zweck entworfenen Items basierte auf Aufgaben, die ursprünglich im MoKoM-Projekt (kurz für: „Entwicklung von qualitativen und quantitativen Messverfahren zu Lehr- und Lernprozessen für Modellierung und Systemverständnis in der Informatik“) erstellt wurden. In diesem von der DFG geförderten Projekt wurde auf Basis eines empirisch validierten Kompetenzmodells ein Testinstrument erstellt, mit dem Kompetenzen von Schülerinnen und Schülern der Sekundarstufe 2 gemessen wurden. Die Autorin war als wissenschaftliche Mitarbeiterin des Projektpartners Universität Siegen an der Entwicklung beteiligt. Der Transfer der Items des MoKoM-Projektes in dieses Forschungsvorhaben gelang, indem fehlerhafte Aufgabenlösungen, die auf einer offensichtlichen Fehlvorstellung basieren, in ein Testitem umgewandelt wurden. Die Aufgabenlösung bestand daraus, die Fehlvorstellung korrekt zu identifizieren und zu beschreiben. Dieses Vorgehen erwies sich als problematisch, da die vorhandenen Items durch die Fokussierung des Projektes auf die Themen Modellierung und Systemverständnis thematisch sehr eingegrenzt waren. Aus diesem Grund erfolgte eine Bearbeitung bzw. Neukonzeption der Items.

Die überarbeiteten Items wurden daraufhin mit Studierenden der Vorlesungen „Didaktik der Informatik I“ und „Didaktik der Informatik II“, die von Prof. Dr. Sigrid Schubert an der Universität Siegen gehalten wurde, erprobt und getestet. Die Ergebnisse wurden in Ohrndorf u. Schubert (2014) veröffentlicht. Ein bedeutsames Fazit aus dieser Erhebung war, dass die Studierenden mehrheitlich Schwierigkeiten bei der Beantwortung fachlich anspruchsvoller Items hatten. So waren die Ergebnisse bei einem Item, in dem die Lösung mit UML modelliert werden musste, sehr schlecht, da die Teilnehmer nicht mit der Syntax vertraut waren. Dies führte zur Entscheidung, den fachwissenschaftlichen Anspruch auf einem geeigneten, aber niedrigen Niveau zu halten und insbesondere die Beherrschung spezifischer Sprachen nicht zu einem Kriterium für die richtige Beantwortung eines Items zu machen.

Auch wenn die in der Haupterhebung dieser Arbeit eingesetzten Testitems nicht mehr mit den ursprünglichen Items aus dem MoKoM-Instrument zusammenhängen, so haben diese doch einen wertvollen Beitrag auf dem Weg zur Entwicklung geleistet. Insbesondere konnten sie einen Anhaltspunkt dafür geben, welche Aufgabentypen für diese Zielgruppe geeignet sind.

Nach der Durchführung der Erhebung zur Identifikation relevanter Konzepte und der Auswertung wurden erste Entwürfe für Items entwickelt, die im Rahmen der ICER Konferenz 2015 in Omaha, USA, vorgestellt und diskutiert wurden (Ohrndorf, 2015). Dies hat zu weiteren Überarbeitungen und Anpassungen des Instruments geführt.

1.4. Gliederung der Arbeit

Die vorliegende Arbeit gliedert sich in sieben Kapitel. Durch die enge Verzahnung, aber thematische Trennung der betrachteten Themenbereiche, ist diese Arbeit nicht streng linear aufgebaut. So fließen zahlreiche Ergebnisse aus den Kapiteln 2 und 3 in die Kapitel 4 bis 6 ein und werden dort erweitert. Eine Übersicht über diese Zusammenhänge ist in Abbildung 1.1 zu finden.

Kapitel 2: Grundlagen und Positionierung

Kapitel 2 stellt die wissenschaftlichen Grundlagen aus der Lernpsychologie und der allgemeinen Didaktik vor, die in der vorliegenden Arbeit relevant sind. Zunächst wird eine Abgrenzung der Begriffe „Fehler“ und „Fehlvorstellung“ vorgenommen, die gleichzeitig auch die Definition darstellt, die diesem Forschungsvorhaben zugrunde liegt. Darauf aufbauend werden Theorien, Klassifikationen und Taxonomien aus der Lernpsychologie vorgestellt, die dabei helfen sollen, Fehler und Fehlvorstellungen zu identifizieren und einzuordnen. Um die Rolle des Wissens über Fehlvorstellungen weiter zu beschreiben, folgt der Perspektivwechsel in die Pädagogik und Didaktik. Dazu sollen zunächst bekannte Topologien und Modelle des Wissens von Lehrenden, mit besonderem Schwerpunkt auf dem Wissen über Fehlvorstellungen, betrachtet werden. Ebenso wichtig ist die Beantwortung der Fragen, wie Lehrende überhaupt dieses Wissen entwickeln und anschließend einsetzen können. Der junge Begriff der „Schülerkognition“ spielt in diesem Kontext eine wichtige Rolle und wird daher vertieft betrachtet. Bevor das Kapitel durch eine Zusammenfassung abgeschlossen

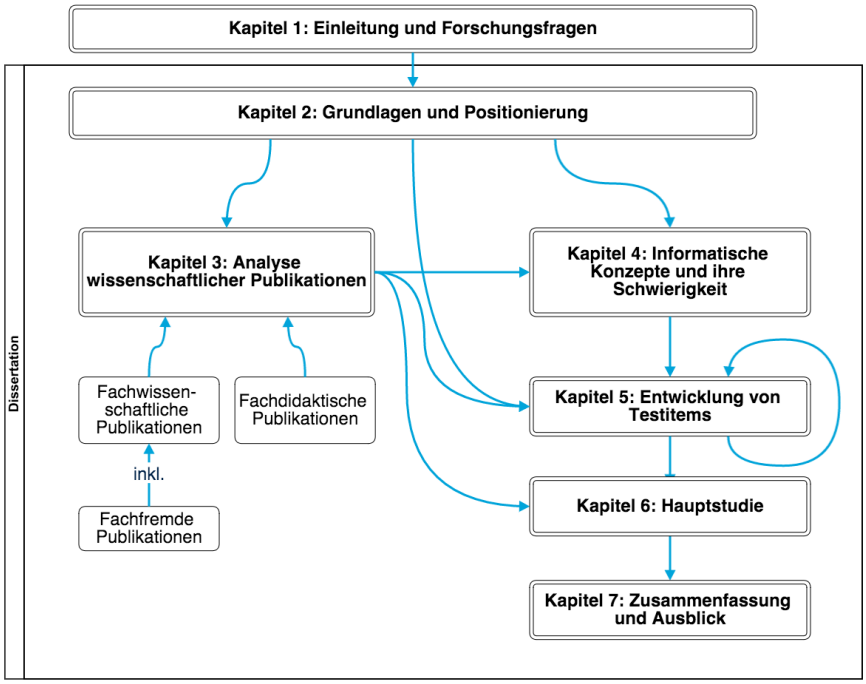


Abbildung 1.1.: Aufbau der Arbeit

wird, wird ein Überblick über die Thematisierung von Fehlvorstellungen in der Ausbildung von Informatiklehrern gegeben.

Kapitel 3: Fehlvorstellungen in der Informatik

Kapitel 3 beschäftigt sich mit der Forschung zu Fehlvorstellungen in der Informatik. Dies beginnt mit einer Analyse der durchgeführten und publizierten Studien im Fachgebiet. Hierzu wird ein Überblick über den Forschungsstand gegeben, der sich zunächst auf die historische Entwicklung der Forschungsvorhaben fokussiert und dabei auch die Entwicklung und Positionierung des Begriffs in der Informatik veranschaulicht. Anschließend geht dieser in eine Vorstellung einzelner Themengebiete über. Darauf folgt eine Analyse der dabei angewandten Methodik. Dies geschieht insbesondere mit dem Ziel, die Ergebnisse zu positionieren und ihre Relevanz und Repräsentativität zu hinterfragen und zu verdeutlichen. Zum Abschluss erfolgt eine Zusammenfassung der Ergebnisse mit besonderer Betrachtung der in den Veröffentlichungen genannten Schlussfolgerungen und Zielsetzungen, sowie eine

Darstellung der Relevanz der Ergebnisse im Kontext der vorliegenden Arbeit. Dieses Kapitel erstellt eine breite Basis, um anschließend die nachfolgenden Kapitel, insbesondere Kapitel 4-6, darauf aufbauen zu können.

Kapitel 4: Identifikation und Bewertung informatischer Konzepte und ihrer Schwierigkeit

In Kapitel 4 werden bekannte Klassifikationsschemata für informatische Themen und Konzepte mit besonderem Schwerpunkt auf den Lerninhalten in Kursen und Vorlesungen für Anfänger vorgestellt. Dies dient insbesondere dazu, Möglichkeiten zur Identifikation und Klassifikation von Fehlvorstellungen in der Informatik aufzuzeigen. Im Anschluss daran erfolgt die Vorbereitung, Zusammenstellung, Durchführung und Auswertung einer Studie, in der grundlegende Konzepte der Informatik und die Häufigkeit damit verbundener Fehlvorstellungen ermittelt werden.

Kapitel 5: Entwicklung von Testitems für die Informatik

Im Mittelpunkt von Kapitel 5 steht die Planung und Konzeption des Testinstruments, mit dem das Wissen über Fehlvorstellungen gemessen werden soll. Auf Basis der in Kapitel 3 und Kapitel 4 gewonnenen Ergebnisse sollen Items generiert werden, die anschließend in der Hauptstudie zum Einsatz kommen werden. Als Grundlage für die Zusammenstellung der Items wird ein Überblick über die vorhandenen Forschungsansätze zu dem jeweiligen Konzept gegeben, um die Relevanz zu verdeutlichen und die Bedeutung des jeweiligen Kontextes zu erklären. Des Weiteren erfolgt eine erste ergebnisorientierte Einschätzung, die später, bei der Auswertung der Studie, wieder einbezogen werden soll.

Kapitel 6: Ergebnisse der Hauptstudie

In Kapitel 6 wird die deskriptive statistische Auswertung der Hauptstudie durchgeführt. Dazu sollen neben der Gesamtauswertung des Instruments und einer Betrachtung demographischer Merkmale auch die einzelnen Items ausgewertet und interpretiert werden. Dies dient dazu, Ergebnisse weiter zu illustrieren und auf diesem Weg Lösungen zu dem Items vorzustellen. Gleichzeitig sollen mögliche Änderungen oder Optimierungen des Instruments festgehalten und dokumentiert werden.

Kapitel 7: Zusammenfassung und Ausblick

In Kapitel 7 findet eine Zusammenfassung der Ergebnisse der Arbeit statt. Es bildet damit einen Abschluss des Forschungsprozesses und fasst zu diesem Zweck zunächst zusammen, inwieweit die zuvor definierten Forschungsziele erreicht wurden. Anschließend wird ein Überblick gegeben über mögliche Anschlussprojekte und offene Fragen in diesem Kontext. Dies schließt insbesondere Ideen und Ansätze ein, wie das erstellte Testinstrument in Zukunft genutzt werden kann.

2. Grundlagen und Positionierung

2.1. Überblick

Dieses Kapitel soll eine Einführung in das Forschungsfeld Fehlvorstellungen geben, indem in Abschnitt 2.2 zunächst eine begriffliche Eingrenzung und Definition vorgenommen wird und in den nachfolgenden Abschnitten darauf aufbauend ein Überblick aus unterschiedlichen wissenschaftlichen Perspektiven gegeben wird. Dazu werden Ergebnissse zu Fehlern und Fehlvorstellungen aus der Lernpsychologie und pädagogischen Psychologie vorgestellt. Diese beziehen sich vornehmlich auf Fehlerarten und die Klassifizierung von Fehlern. Sie verdeutlichen, wie einfache Fehler von Fehlvorstellungen abzugrenzen sind. Anschließend erfolgt ein Wechsel zur Perspektive der Lehrenden, indem Fehler und Fehlvorstellungen in ihrem Wissen und Handeln verortet werden und Theorien und Modelle über die Entwicklung präsentiert werden. Dies ist eine Hinführung zu der in Kapitel 3 folgenden Analyse und Bewertung von Fehlvorstellungen in der Informatik.

Die vertiefte Betrachtung von Fehlern mag nach der Abgrenzung der Fehlvorstellungen von ebendiesen im vorherigen Kapitel zunächst irritierend sein und nicht zielführend wirken. Im Laufe der nächsten Abschnitte sollte aber deutlich werden, dass eine Trennung der beiden Forschungsgebiete nicht möglich ist. Insbesondere in den in Abschnitt 2.3 vorgestellten Fehlertaxonomien finden sich immer wieder Verweise auf Fehlvorstellungen, die nicht ignoriert werden sollten. Des Weiteren erfolgt insbesondere in den früheren Publikationen keine explizite Abgrenzung zwischen Fehlern und Fehlvorstellungen statt - Fehlvorstellungen werden vielmehr als spezieller Typ von Fehlern betrachtet.

2.2. Fehler und Fehlvorstellungen

Fehler und Fehlvorstellungen sind eng miteinander verknüpft und werden, wenn sie in einem nichtwissenschaftlichen Kontext erwähnt werden, selten voneinander abgegrenzt. Eine solche Abgrenzung ist aber nötig, um die tatsächlichen Eigenschaften von Fehlvorstellungen zu verdeutlichen und ihnen mehr Raum abseits einer binären „falsch oder richtig“ Klassifikation zu geben.

Fehler werden im Duden als „etwas, was falsch ist, vom Richtigen abweicht; Unrichtigkeit“ definiert. Dass diese Einschätzung zwar in der Alltagssprache ausreichend und allgemein anerkannt ist, sowie verwendet wird, ist unbestreitbar. Einer wissenschaftlichen Betrachtung von Fehlern hält sie indes nicht stand. Nach Oser u. Spychinger (2005) ist ein Fehler ein Sachverhalt oder Prozess, der von der Norm abweicht. Dies ist der zuvor genannten Definition sehr ähnlich. Es ergibt sich allerdings zwangsläufig die Frage, wie eine solche Norm definiert ist, die vermutlich nur individuell für jeden einzelnen Fehler zu beantworten ist. Dies verdeutlicht, dass Fehler grundsätzlich situationsbezogen bewertet und analysiert werden müssen. Weitere Herangehensweisen hierzu werden im folgenden Abschnitt 2.3 erläutert.

Die Definition eines Fehlers lässt sich nicht auf Vorstellungen von Lernenden übertragen, da diese nicht zwangsläufig fehlerhaft sind, sondern auch lediglich unvollständig sein können. Dies ist eine Eigenschaft, die die Fehlvorstellung vom Fehler deutlich abgrenzt. Zudem sind Fehlvorstellungen weniger offensichtlich als ein Fehler und können auch über einen langen Zeitraum hinweg existieren, ohne aufzufallen. Fehlvorstellungen sind daher in der Forschung ein beliebtes Thema, da ihre Identifikation schwer und die Entstehung und Entwicklung häufig nicht offensichtlich ist.

Smith u. a. beschreiben den Ursprung der Forschung zu Fehlvorstellungen wie folgt:

> „The idea that students develop misconceptions lies at the heart of much of the empirical research on learning mathematics and science of the last 15 years. Following Piaget's repeated demonstrations that children think about the world in very different ways than do adults, educational researchers in the late 1970s began to listen carefully to what students were saying and doing on a variety of subject-matter tasks.“ (Smith u. a., 1994)

2.2.1. Kritik am Begriff der Fehlvorstellung

Der Begriff der „Fehlvorstellung“ wurde mehrfach kritisiert, da er suggeriert, dass diese vorhandenen Vorstellungen grundsätzlich fehlerhaft und dadurch negativ sind. Dass Fehlvorstellungen auch nur unvollständig sein oder auf einer falschen Abstraktionsebene existieren können, kann missverstanden werden.

Diethelm u. a. (2012) beschreiben im Kontext einer Erhebung zum Wissen von Schülern über das Internet die Verwendung der Begriffe „misconception“ und „preconception“ wie folgt:

> „Following this model where each corner has to be taken into account equally for the design and arrangement of lessons and courses, one cannot speak about misconceptions because „mis“ already judges on what might be wrong or worthwhile and what not. Here, one can only speak about preconceptions of the students and regard them as fruitful in every case, suitable to the scientific view or not.“ (S. 68 Diethelm u. a., 2012)

Diese Erklärung basiert darauf, dass der Vorsilbe „mis-“ eine negative Bedeutung zugesprochen wird, die der tatsächlichen Definition einer Fehlvorstellung nicht gerecht wird. Dies ist allerdings gleichzeitig eine Interpretation des Begriffes, die sehr spitzfindig ist. Dadurch, dass eine Vorstellung nicht der wissenschaftlichen Sicht entspricht, ist sie in jedem Fall nicht richtig - entweder in der Form, dass tatsächlich ein Aspekt des jeweiligen Themas oder Konzeptes fehlt, sie also unvollständig ist, aber auch, indem dieses Wissen zwar zu diesem Zeitpunkt anwendbar ist, aber nicht fortgeführt und erweitert werden kann. Die Vorsilben „mis“ oder „Fehl“ definieren also insbesondere, dass eine Vorstellung vorhanden ist, die in irgendeiner Weise von der Norm oder den Erwartungen abweicht.

Zugleich kann der Begriff des Präkonzeptes (bzw. engl. preconcept) falsch interpretiert werden und gibt eine unvollständige Information über die Entstehung von Fehlvorstellungen. Er legt nahe, dass diese Konzepte „zuvor“ gebildet werden, d. h. im Regelfall vor Beginn einer strukturierten und regelmässigen Ausbildung. Dass dies nicht der Fall ist und Fehlvorstellungen auch durch Faktoren und Aspekte in der Ausbildung entstehen können, wird dabei vernachlässigt. Aus diesem Grund wird im folgenden der Begriff „Fehlvorstellung“ konsequent verwendet.

2.3. Fehler und Fehlvorstellungen in der Lernpsychologie

Die zuvor genannten Beschreibungen sind teilweise komplex, häufig auch simpel und anhand von Alltagsvorstellungen nachvollziehbar, geben aber gleichzeitig auch wenig Information darüber, wo exakt der Unterschied zwischen einem Fehler und einer Fehlvorstellung liegt. Im Folgenden soll daher zunächst ein Überblick über die Fehlerforschung in der Lernpsychologie gegeben werden. Als Erstes seien Klassifizierungen aus der Lehr-Lerntheorie genannt, die fächerübergreifend angewandt werden können. Dabei ist anzumerken, dass diese nicht zwingend miteinander vereinbar, in einzelnen Fällen sogar widersprüchlich sind. Daran anschließend sollen konkrete Klassifikationsschemata (Taxonomien) für Fehler und Fehlvorstellungen vorgestellt werden, die zumeist eine fachspezifische Beurteilung ermöglichen sollen. Da es hier nur sehr vereinzelte Ansätze in der Informatik gibt, werden Ergebnisse aus den Naturwissenschaften und der Mathematik herangezogen. Diese werden anschließend zum Schwerpunkt dieser Arbeit, den Fehlvorstellungen, überführt.

2.3.1. Begriffe und ihre Verwendung

Als intuitive Klassifikation lässt sich die auch in die Alltagssprache eingegangene Unterscheidung zwischen Flüchtigkeitsfehlern und systematischen Fehlern (vgl. u. a. Radatz, 1979; Prediger u. Wittmann, 2009; Bromme, 1997) nennen, die häufig in der Literatur aufgegriffen wird. Der Flüchtigkeitsfehler ist auf Faktoren zurückzuführen, die unabhängig vom Wissen des Lernenden sind, wie z. B. fehlende Aufmerksamkeit oder Ablenkung. Der Lernende kann diesen Fehler sofort korrigieren, sobald er darauf hingewiesen wird. Dementsprechend ist ein Flüchtigkeitsfehler auch nicht auf den Lernprozess zurückführen. In der Informatik könnte dies ein vergessenes Semikolon oder eine nicht geschlossene Klammer in einem Programmcode sein.

Ein systematischer Fehler tritt hingegen bei Aufgaben desselben Typs wiederholt auf, was zudem ein starkes Indiz dafür ist, dass die Ursache nicht in der Aufgabenstellung liegt. Dem Lernenden ist die Fehlerhaftigkeit nicht bewusst und er kann gegebenenfalls auch seinen falschen Lösungsprozess erläutern und begründen. Er ist somit im Regelfall sicher, dass seine Lösung richtig ist oder schätzt sie als konsistent ein. Dies stellt den für den Lernprozess deutlich interessanteren Fehler dar, da eine Korrektur nicht selbstständig erfolgen kann und eine Reaktion des Lehrenden nötig ist.

Die Ursache für einen Fehler lässt sich sowohl auf syntaktischer als auch auf semantischer Ebene finden. Ein syntaktischer Fehler ist auf die falsche Anwendung von Regeln zurückzuführen. Ein semantischer Fehler basiert hingegen auf einem falschen Verständnis der Bedeutung. Beide Fehlerarten werden voneinander begünstigt, wobei syntaktische Fehler häufig durch Fehler in der Semantik entstehen (vgl. Prediger u. Wittmann, 2009). Für die Informatik ist an dieser Stelle anzumerken, dass ein Syntaxfehler nicht zwangsläufig ein syntaktischer Fehler sein muss.

Prediger u. Wittmann (2009, S. 5) nennen für die Unterscheidung von syntaktischen und semantischen Fehlern das folgende Beispiel aus der Mathematik:

Beispielaufgabe zu syntaktischen und semantischen Fehlern

Aufgabe: Ein Kilogramm Mandarinen kostet 3,25 Euro. Kerstin will sich 0,5 kg kaufen.Was muss sie zahlen?

Lillys Lösungsweg:
$3,25 * 0,5 = 0,125 \approx 0,13$
Sie muss 0,13 € zahlen.

Hier kann es sich einerseits um einen syntaktischen Fehler handeln, der durch eine falsche Anwendung der Rechenregeln entsteht oder um einen semantischen Fehler, der auf ein falsches Stellenwertverständnis zurückzuführen ist. Das zeigt, dass es auf Basis einer einzelnen Aufgabenlösung schwer ist, eine Fehlerursache zu identifizieren. Dazu muss entweder der Lösungsweg nachvollzogen werden (d. h. im Regelfall muss dieser durch den Bearbeiter erklärt werden) oder durch mehrere falsche Lösungen eine Logik oder Systematik erkannt werden, die zum Fehlermuster führt. Wenn dieses vorhanden ist, handelt es sich nicht um einen Flüchtigkeitsfehler.

Fehler können auch nach der Form der Informationsaufnahme und -verarbeitung klassifiziert werden. Radatz (1979) zählt dazu für die Mathematikdidaktik Sprachschwierigkeiten, Schwierigkeiten beim räumlichen Vorstellungsvermögen (*errors in obtaining spatial information*), Defizite hinsichtlich erforderlicher Fähigkeiten, Wissen und Konzepte, falsche Assoziationen und die Anwendung nicht zutreffender Regeln auf. Diese Fehler sind einerseits spezifisch für das Fach (insbesondere Schwierigkeiten beim räumlichen Vorstellungsvermögen), aber auch generisch. Insbesondere Sprachschwierigkeiten führen in nahezu jeder Fachdisziplin zu Fehlern, indem Aufgabenstellungen falsch interpretiert oder Aufgabenlösungen nicht korrekt ausformuliert werden.

Klassifizierung	Einordnung	Autor
Flüchtigkeitsfehler und systematische Fehler	allgemein	Radatz (1979); Bromme (1997); Prediger u. Wittmann (2009)
Syntaktisch und Semantisch	allgemein	Prediger u. Wittmann (2009)
Informationsaufnahme	allgemein	Radatz (1979)
Informationsverarbeitung	allgemein	Radatz (1979)
Kontext inhaltlich	fachspezifisch	unbekannt
Kontext kognitiv	allg. und fachsp.	Mindnich u. a. (2008)
Phasen Aufgabenbearbeitung	allg. und fachsp.	Müller (2003)

Tabelle 2.2.: Übersicht binärer Klassifizierungsschemata für Fehler

Müller unterscheidet zwischen Fehlern, die in der in der Planungsphase einer Aufgabe auftreten und jenen, die erst in der Ausführungsphase entstehen. Die Betrachtungen stammen aus der Physikdidaktik, lassen sich aber insbesondere auf die Mathematik, die Naturwissenschaften und auch die Informatik übertragen. Ein Planungsfehler liegt vor, wenn falsche Konzepte gewählt werden, die Aspekte außer Acht lassen oder schlicht nicht geeignet sind:

> „Ein Beispiel für einen sehr elementaren Planungsfehler aus der Physik (Bereich Strahlenoptik) ist, bei der Frage nach der Erklärung der Kurzsichtigkeit aus optischer Sicht nur auf die im Vergleich zu Normalsichtigen zu große Länge des Augapfels hinzuweisen; tatsächlich kommt es ja auf die Abstimmung der Länge des Augapfels und der Gesamtbrennweite des Auges an, und der Fehler besteht darin, dass nicht beide Faktoren berücksichtigt werden.“ (Müller, 2003, S. 11)

Diese Fehlerart wird in der Alltagssprache als „falscher Ansatz“ bezeichnet, d. h. basierend auf der Aufgabenstellung wird der falsche Lösungsweg gewählt. Der Lösungsweg als solcher kann richtig ausgeführt sein, ergibt allerdings keine Lösung für die Aufgabe.

Ein Ausführungsfehler setzt hingegen eine richtige Planung voraus, führt aber beispielsweise durch Zeitmangel oder fehlende Aufmerksamkeit bei der Bearbeitung der Aufgabe nicht zum Ziel. Die richtige Lösung kann hier wiederum durch einen Flüchtigkeitsfehler verhindert werden. Auch kann ein Gedächtnisfehler auftreten, bei dem z. B. ein Arbeitsschritt vergessen wird oder Zwischenergebnisse falsch übertragen werden.

Die verschiedenen Systematiken, die noch einmal in Tabelle 2.2 zusammengefasst werden, zeigen auf, dass keine eindeutige Form der Klassifizierung existiert. Neben grundlegend verschiedenen Ansätzen ergeben sich häufig Überschneidungen, die eine direkte klassifikationsübergreifende Zuordnung und damit auch eine Abgrenzung voneinander unmöglich machen. Daraus ergibt sich zwangsläufig die Frage, welchen Nutzen eine Klassifizierung hat und wie im Einzelfall entschieden werden kann, was eine geeignete Klassifikation ist. Die Klassifikation bietet zwar die Möglichkeit, Fehler einzuschätzen und zu sammeln, bietet aber wenige bis keine Ansätze dazu, wie letztendlich in der Praxis mit den klassifizierten Fehlern umgegangen werden soll. Mindnich u. a. (2008) nennen als Nutzen die Möglichkeit, dem Lernpotential von Fehlersituationen auf die Spur zu kommen und diese bestimmten Schülertypen und Unterrichtsphasen zuzuordnen. Sie können insbesondere dazu genutzt werden, Fehlern Ursachen zuzuordnen und daraus mögliche Handlungskonzepte abzuleiten. Auf diese wird in Abschnitt 2.4 vertieft eingegangen.

2.3.2. Taxonomisierung von Fehlern

Wie zuvor dargestellt, können Fehler mithilfe von Taxonomien eingeordnet und klassifiziert werden. Bisher wurden nur simple Klassifizierungen präsentiert, die eine einfache Ursachenunterscheidung vornehmen. Neben diesen existieren allerdings auch umfangreiche Taxonomien, die komplexe Phasen der Aufgabenbearbeitung oder sogar fachspezifische Arbeitsschritte repräsentieren. Einige dieser Taxonomien, insbesondere jene, die für die Informatik geeignet scheinen, sollen an dieser Stelle in einer Auswahl vorgestellt werden.

Klassifizierung von Fehlern nach Mindnich

Die von Mindnich u. a. (2008) durchgeführte Pilotstudie zur Typisierung von Fehlern und Fehlersituationen basiert auf der Analyse von Unterrichtsvideos, die im Rechnungswesenunterricht bei drei Lehrkräften entstanden. Die Videos wurden anhand einer Kodieranleitung von zwei Kodierern analysiert und den fünf Fehlerarten *Reproduktion*, *Verständnis*, *Anwendung*, *Informationserzeugung* und *Sonstige* zugeordnet, die an die „cognitive process dimensions“ nach Anderson und Krathwohl (Anderson u. Krathwohl, 2001) angelehnt sind. Eine Übersicht der Fehlerarten mit einer kurzen Erklärung befindet sich in Tabelle 2.4. In der Auswertung zeigte sich, dass die Mehrheit der Fehler den Verständnisfehlern zuzuordnen sind, während Fehler bei der Informationserzeugung nicht auftraten.

Stufe	Beschreibung	
1	Reproduktionsfehler	Fehler beim Abruf von bereits gelernten Inhalten
2	Verständnisfehler	Fehler bei dem Bedeutungsgehalt oder der Beziehung zwischen Wissenselementen
3	Anwendungsfehler	Fehler bei der Anwendung von Wissen auf neue Situationen
4	Analyse, Evaluation und Kreation	Fehler bei der Informationserzeugung
5	sonstige Fehler	Fehler, die sich nicht zu den vorhergehenden Fehlerarten zuordnen lassen

Tabelle 2.4.: Fehlerarten nach Mindnich u. a. (2008), basierend auf der Klassifizierung nach Anderson u. Krathwohl (2001)

Newman's Error Analysis (NEA)

Die Newman's Error Analysis (NEA) ist ein Klassifikationsschema, mit dem Fehler bei der Lösung von mathematischen Textaufgaben analysiert werden können. Dieser Aufgabenklasse kommt eine besondere Bedeutung zu, da die Lesekompetenz als Schlüsselqualifikation anzusehen ist und damit die erfolgreiche Bearbeitung elementar beeinflusst. Des Weiteren ist der Schwierigkeitsgrad einer solchen Aufgabe stark abhängig von ihrer sprachlichen Formulierung. So kann die Beantwortung stark vereinfacht werden, wenn in der Aufgabenstellung explizite Hinweise zum Lösungsweg vorhanden sind. Ein Problem kann aber auch darin liegen, dass Texte missverständlich oder unpräzise sind. Dies führt dazu, dass es bei der Analyse der bei Textaufgaben auftretenden Fehlern einer Differenzierung der benötigten Kompetenzen bedarf. Um den Bearbeitungsprozess exakt abzubilden, erstellte Newman (vgl. White, 2009) die NEA (s. Tabelle 2.6). Die ursprüngliche Konzeption der Klassifikation sieht vor, dass diese in Form eines Interviews erfolgt, in denen der Schüler seinen Lösungsprozess schrittweise beschreibt.

Stufe	Beschreibung
1	Please read the question to me. If you don't know a word, leave it out. (Reading)
2	Tell me what the question is asking you to do. (Comprehension)
3	Tell me how you are going to find the answer. (Transformation)
4	Show me what to do to get the answer. "Talk aloud" as you do it, so that I can understand how you are thinking. (Proccess Skills)
5	Now, write down your answer to the question. (Encoding)

Tabelle 2.6.: Newman's Error Analysis

Ein großer Nachteil der NEA ist darin zu sehen, dass die individuelle Fehleranalyse sehr aufwändig ist. Sie ist in dieser Form daher schwerpunktmässig als Mittel zur Einzelanalyse lernschwacher Schüler anzusehen und wird als solches eingesetzt (White, 2009). In diesem Feld ist sie auch als zuverlässiges Werkzeug anzusehen, mit dem Fehlvorstellungen identifiziert werden können.

Structure of Observed Learning Outcome (SOLO)

Biggs (Biggs, 1979; Biggs u. Collis, 1982) entwickelte die Structure of Observed Learning Outcome (SOLO) Taxonomie, die die wachsende Komplexität des Wissens von Lernenden im Laufe des Lernprozesses beschreibt. Im Gegensatz zu anderen Taxonomien, betrachtet sie die Antwort des Lehrenden inhaltlich. Ein Nachteil, der sich daraus ergibt, sind fehlende Richtlinien bei der Kategorisierung, wie sie z. B. bei der Taxonomie nach Bloom gegeben sind (Fuller u. a., 2007).

Er definiert die einzelnen Stufen wie folgt (zitiert nach Biggs, 1979):

1. *Pre-structural.* The response has no logical relationship to the display, being based on inability to comprehend, tautology or idiosyncratic relevance.
2. *Uni-structural.* The response contains one relevant item from the display, but misses others that might modify or contradict the response. There is a rapid closure that oversimplifies the issue.
3. *Multi-structural.* The response contains several relevant items, but only those that are consistent with the chosen conclusion are stated. Closure is selective and premature.

4. *Relational.* Most or all of the relevant data are used, and conflicts resolved by the use of a relating concept that applies to the given context of the display, which leads to a firm conclusion.
5. *Extended abstract.* The context is seen only as one instance of a general case. Questioning of basic assumptions, counter examples and new data are often given that did not form part of the original display. Consequently a firm closure is often seen to be inappropriate.

Die einzelnen Schritte ähneln der Entwicklungshierarchie nach Piaget (vgl. Krapp u. Weidenmann, 2009), unterscheiden sich aber fundamental, da sie nur jeweils eine einzelne Antwort bzw. Lösung betrachten. Wo hier Fehlvorstellungen einzuordnen sind, lässt sich nicht klar bestimmen. Lister u. a. assoziieren die erste Stufe mit der Existenz einer Fehlvorstellung oder einem Präkonzept:

> „In terms of reading and understanding a small piece of code, a student who gives a prestructural response is manifesting either a significant misconception of programming, or is using a preconception that is irrelevant to programming.“ (Lister u. a., 2006b)

Durch den Wechsel in die nächsthöhere Stufe, dürften einerseits Präkonzepte ersetzt werden, andererseits schließt auch dies die Existenz von Fehlvorstellungen nicht aus. Dies ist erst in der vierten Stufe zu erreichen, in der das Wissen auch angewandt, differenziert und diskutiert werden kann. Die SOLO-Taxonomie ist unabhängig von der Disziplin und lässt sich somit uneingeschränkt auf die Informatik übertragen.

Jimoyiannis (2011) hat nach Vorbild der SOLO-Taxonomie spezifisch für die Informatik die folgende Kategorisierung für das Verständnis für Code entwickelt:

1. *Prestructural*: This is the least sophisticated type of response a student may give to a programming task. A prestructural response manifests either a significant misconception of programming or a preconception that is irrelevant to programming. The student lacks knowledge of programming constructs and approaches the task under study in a non appropriate or unrelated way.
2. *Unistructural*: This is a response where the student manifests a correct grasp of some but not all aspects of the programming problem. The student has a partial understanding and one or few aspects are picked up and used effectually. In many cases, students make what is called an “educated guess” (Lister u. a., 2006a). For

example, the student describes the functioning of a part (one or two lines) of the code.

3. *Multistructural*: The student focuses on several relevant aspects but does not manifest an awareness of the relationships between them or the whole. According to Lister u. a., the student fails „to see the forest for the trees“. For example, a student may provide a line-by-line description of the code or execute the code by hand and arrive at a final value for a particular variable. However, he is not able to see the code as a single coherent construct.
4. *Relational*: This level corresponds to what is normally meant by adequate understanding of the topic under study. The student makes sense of the various aspects of the topic, integrates the parts of the problem into a coherent code structure, and uses this structure to solve the problem. According to Lister u. a. (2006a), “the student sees the forest”. A relational response is the most sophisticated type of response a student may give and may be either correct or incorrect. For example, a student is able to describe the function performed by a particular code segment without hand executing. The student may infer that the code counts the number of elements in an array which are greater than a particular value.
5. *Extended Abstract*: In this highest SOLO level, the student response goes beyond the particular problem to be solved, and links the problem to a broader context. The student is able to extrapolate, to develop higher order principles and extend the topic to wider application areas. For example, a possible extended abstract response may be a comment that the code will only work for arrays that are sorted (Lister u. a., 2006a). While this is a very interesting level to study, the tasks designed for the present study do not aimed at promoting students' performance in the extended abstract level.

Er ordnet in seiner Studie die Schülerantworten zu sechs verschiedenen Programmieraufgaben in dieses Klassifizierungsschema ein. Hier zeigte sich, dass etwa die Hälfte der Schüler Wissen auf den ersten drei Stufen hat. Dies ist jedoch hauptsächlich auf die Aufgabenstellung in dieser Studie zurückzuführen, die keine vertiefte Betrachtung auf einer höheren Ebene zuließ. Jimoyiannis sieht die SOLO Taxonomie als ein sehr effizientes Instrument, um die Leistung von Lernenden in Einführungskursen in die Programmierung zu bewerten und schlägt einen weiterführenden Einsatz, auch in internationalen Studien, vor.

Thompson (2007) setzte ebenfalls die SOLO-Taxonomie ein und nutzte die Kriterien um die Notenvergabe in Programmierprojekten zu definieren. Diese sollten Studierenden dabei helfen einzuschätzen, welche Leistung für welche Note verlangt wird. Er kann nach dem Einsatz ein positives Fazit ziehen, stellt aber auch die Frage, inwieweit sich die SOLO-Taxonomie so anpassen lässt, dass sie vielseitig und auch bei kleineren Projekten eingesetzt werden kann.

2.3.3. Fehlvorstellungen

Die bisher genannten Ergebnisse zeigen, dass häufig keine klare Trennung von Fehlvorstellungen und Fehlern vorgenommen wird, die der Definition in Abschnitt 2.2 entsprechen. Die Fehlerforschung betrachtet häufig Zusammenhänge und Klassifizierungen, die auch Fehlvorstellungen einschließen, erwähnt diese jedoch nicht explizit. So haben z. B. die unteren Stufen der SOLO-Taxonomie auch Eigenschaften, die auf eine Fehlvorstellung schließen lassen. Viele Fehlerursachen lassen sich demnach in das Gebiet der Fehlvorstellungen einordnen, selbst wenn sie in den jeweiligen Forschungsergebnissen konsequent als Fehler klassifiziert werden. Dies bestätigt, dass Fehlvorstellungen einer anderen Ebene zuzuordnen sind als Fehler: sie können einerseits eine Ursache für Fehler sein, müssen dies aber andererseits nicht, wenn sie bei der Aufgabenbearbeitung nicht hinderlich sind.

Eines der ersten Modelle zu Fehlvorstellungen in naturwissenschaftlich-technischen Fachgebieten ist die Arbeit von Brown u. Burton (1978), die sich mit grundlegenden mathematischen Rechenoperationen befasst. Diese enthält ein Modell, das die Identifikation und Interpretation von Fehlvorstellungen erleichtern soll. Dieser Forschungsansatz wird später um eine „repair theory“ erweitert (Brown u. VanLehn, 1980). Diese besagt, dass ein Schüler sein vorhandenes prozedurales Wissen zur Lösung von Aufgaben einsetzt. Wenn er allerdings in eine Sackgasse („impasse“) gerät und dieses nicht anwenden kann, versucht er dieser zu entkommen, indem er bewährte Subprozeduren nutzt und darauf aufbauend eine Lösungstheorie bildet („repair“). Die Fehlvorstellung wird durch die (falsche) Anwendung vorhandenen Wissens auf ein neues Problem generiert. Dass sich diese Theorie nur auf einen Teil der Fehlvorstellungen anwenden lässt, ist durch den beschriebenen Entstehungsprozess offensichtlich. Gleichzeitig erscheint es auch schwer, einen solchen Prozess zu umgehen, da präventiv vorgegangen werden muss, um mögliche Fehlvorstellungen zu vermeiden. Dies ist insbesondere bei informellen Lernprozessen unmöglich.

Es existieren weitere Begriffe, die teilweise auch synonym oder ergänzend zum Begriff „Fehlvorstellungen“ gebraucht werden. Erwähnenswert ist hier das bereits

zuvor erwähnte Präkonzept, das Konzepte und Vorstellungen von Lernenden beschreibt, die vor Beginn des wissenschaftlich gestützten Lernprozesses gebildet werden. Damit wird eine zeitliche Einordnung vorgenommen, die insofern von Bedeutung ist, da so beschrieben werden kann, wo Lernende zu Beginn ihrer Ausbildung „abgeholt“ werden müssen und worauf dabei Rücksicht genommen werden muss. Ein Beispiel hierfür sind physikalische Gesetzmässigkeiten, die Kinder spielerisch beobachten, z. B. die Erdanziehungskraft durch ein fallendes Spielzeug. Präkonzepte sind nicht grundsätzlich mit dem Wissen gleichzusetzen, das vor Beginn der schulischen Ausbildung erworben wird. Wissen in dieser Phase kann auch strukturiert und den wissenschaftlichen Ansprüchen genügend erworben werden. Ein Beispiel hierfür sind beispielsweise Lehrbücher, die ein Phänomen ausreichend und umfassend beschreiben. Die Entstehung von Präkonzepten beschränkt sich nicht auf den Lernprozess von Kindern: auch Erwachsene werden häufig mit Konzepten konfrontiert, die sie wissenschaftlich nicht nachvollziehen und begründen können. Ein Präkonzept entsteht dann, wenn Konzepte ohne vertieftes Wissen aufgenommen werden und falsche oder unvollständige Hypothesen durch Beobachtungen oder Schlussfolgerungen entwickelt werden.

Ein weiteres Problem, das häufig in der expliziten Forschung zu Fehlvorstellungen betrachtet wird, ist falsches Vorwissen (entsprechend einer Fehlvorstellung) und mit neuem Wissen ergänzt oder verknüpft werden soll, aber mit diesem unvereinbar ist. Dies führt zwangsläufig zu einer Pause im Lernprozess, da eine Seite zunächst angepasst werden muss: entweder muss die Fehlvorstellung korrigiert werden oder fälschlicherweise das neu zu erlernende so angepasst werden, dass es zum vorhandenen Wissen passt. Im Allgemeinen tendieren die Lernenden zur letzteren Variante, da sie an das Vorwissen gewöhnt sind, es als valide betrachten und vielleicht auch schon praktisch erfolgreich angewandt haben, ohne die Fehlvorstellung zu erkennen (Merenluoto, 2004). Auf diese Weise vergrößert sich die Fehlvorstellung und verhindert den Wissenserwerb. In jedem Fall benötigt eine Anpassung des Wissens, sei es des neu zu erlernenden oder des bereits vorhandenen, viel Zeit.

Ein in diesem Kontext beobachtetes Phänomen ist außerdem, dass Lernende zwar das wissenschaftlich richtige Konzept erlernen und es korrekt anwenden können, aber nicht in der Lage sind, es in den Alltag zu integrieren bzw. es auf andere Situationen zu übertragen. Hier liegt somit eine starke Fixierung auf die direkte Anwendung des Erlernten vor. So können z.B. bekannte und bereits gelöste Aufgaben richtig beantwortet werden, indem der Lösungsweg adaptiert und leicht angepasst wird. Wenn die Aufgabe in abgeänderter Form gelöst oder das Gelernte in einem anderen Kontext angewandt werden soll, ist dies nicht erfolgreich. Auch hier liegt

eine Fehlvorstellung vor, da das zu erlernende Konzept als solches falsch oder unvollständig erlernt wurde.

Die Ursache für die Entwicklung von Fehlvorstellungen ist tief im menschlichen Lernverhalten verankert. Das grundsätzliche Lernen besteht aus der Beobachtung der Umwelt, dem Verknüpfen mit bereits vorhandenem Wissen und der Mustererkennung. Dass aus diesem Dreieck auch falsche Konzepte hervorgehen können ist offensichtlich und unvermeidbar.

Aufbauend auf den zuvor genannten Ergebnissen soll an dieser Stelle eine Definition angegeben werden, um die Verwendung des Begriffes „Fehlvorstellung“ im Kontext dieser Arbeit zu beschreiben:

> Fehlvorstellungen sind Vorstellungen, die den fachwissenschaftlich korrekten Konzepten widersprechen oder von ihnen abweichen.

2.4. Fehler und Fehlvorstellungen in der Didaktik und Pädagogik

Im Folgenden soll der Forschungsstand zu Fehlern und Fehlvorstellungen aus Sicht der Didaktik und Pädagogik vorgestellt werden. Im Gegensatz zum vorhergehenden Abschnitt liegt hier die Perspektive speziell auf dem Aspekt der Lehre. Es erfolgt somit der Schritt von den Fehlvorstellungen zum Wissen über Fehlvorstellungen und dem Umgang mit Fehlvorstellungen, dem Schwerpunkt der vorliegenden Arbeit.

2.4.1. Definition und Entwicklung des fachdidaktischen Wissens

Dass für die erfolgreiche Lehre mehr als ein ausgeprägtes Fachwissen nötig ist, ist unbestreitbar. Jeder Lernende kennt „gute“ und „schlechte“ Lehrer aus der Schule oder auch Dozenten an Universitäten. Was gute Lehrer gemein haben ist die Fähigkeit, Wissen auf einem kognitiven Niveau zu vermitteln, das angemessen für die Lernenden ist und zu erkennen, wenn diese überfordert sind oder konkrete Probleme und Verständnisschwierigkeiten auftreten. Was hingegen genau eine Person ausmacht, die dies erfolgreich kann, ist schon seit vielen Jahren Gegenstand von Forschungsarbeiten.

Shulman (1986, 1987) stellte eine Topologie des Lehrerwissens auf, die definieren soll, welches Wissen ein Lehrer braucht und inwieweit sich dieses Wissen von dem eines Fachwissenschaftlers unterscheidet. Sie gilt in der (Fach-)Didaktik als allgemein akzeptiertes und vor allem auch validiertes Konzept zur Strukturierung des Lehrerwissens. Shulman formulierte sieben Kompetenzen, von denen drei heute als zentrale Bestandteile des Professionswissens von Lehrern gelten (vgl. Krauss u. a., 2008a, S. 181):

- content knowledge (Fachwissen)
- pedagogical content knowledge (fachdidaktisches Wissen[1]), pedagogical knowledge (pädagogisches Wissen)
- curricular knowledge (Curriculares Wissen)

Dabei stellt die Pedagogical Content Knowledge (PCK) den entscheidenden Part dar, da sie das Wissen ist, das ein guter Fachwissenschaftler benötigt um auch ein guter Lehrender zu sein. Shulman (1987) selbst beschreibt die PCK wie folgt:

> „It represents the blending of content and pedagogy into an understanding of how particular topics, problems, or issues are organized, represented, and adapted to the diverse interests and abilities of learners, and presented for instruction. Pedagogical content knowledge is the category most likely to distinguish the understanding of the content specialist from that of the pedagogue."

Eine Kompetenz, die sowohl fachspezifische, als auch fachübergreifende Aspekte beinhaltet und als solche in das fachdidaktische Wissen nach Shulman eingeordnet werden kann, ist die diagnostisches Kompetenz. Sie beschreibt die Fähigkeit, Lernende und ihre Leistung korrekt zu beurteilen. Dies bezieht sich nicht darauf, den Leistungsstand der Lernenden mit Hilfe von Tests oder Befragungen zu messen, sondern darauf, Lernende vor Beginn und während des Lernprozesses richtig einzuschätzen und zu beurteilen. Darunter fällt auch die Identifikation von Fehlvorstellungen. Diese Kompetenz wird dementsprechend dem PCK-Modell zugeordnet (Kunter u. Pohlmann, 2009).

Auch wenn die Kategorisierung nach Shulman weit verbreitet ist und oft als grundlegende Darstellung des Lehrerwissens herangezogen wird, stellt sich die Frage, ob eine eindeutige Aufteilung der unterschiedlichen Bereiche überhaupt möglich ist.

[1] Diese Übersetzung ist nicht eindeutig, da sie die anglo-amerikanische „curriculum tradition" und die davon unabhängig entwickelte deutsche Fachdidaktik vermischt (vgl. Fensham, 2002; Kansanen, 2009). Da dieser Begriff gemeinhin synonym verwendet wird (vgl. Baumert u. Kunter, 2006), wird er in dieser Arbeit übernommen.

Zweifellos ist das fachdidaktische Wissen abhängig vom Fachwissen: um Inhalte vermitteln zu können, muss der Lernende diese auch selbst vollständig verstanden haben. So ergeben sich zwangsläufig starke Abhängigkeiten zwischen fachdidaktischem Wissen und Fachwissen. Dies wird auch aus Shulmans Beschreibung der PCK deutlich:

> „A second kind of content knowledge is pedagogical knowledge, which goes beyond knowledge of subject matter per se to the dimension of subject matter knowledge for teaching.“ (Shulman, 1986)

Dies bedeutet nicht zwangsläufig, dass das fachdidaktische Wissen stetig mit dem Fachwissen anwächst. Ganz im Gegenteil kann auch die Hypothese aufgestellt werden, dass ein zu tiefes Fachwissen die Fähigkeit, dieses angemessen auf einem niedrigen kognitiven Niveau zu vermitteln, behindert. Diese Frage wurde vielfach in den Fachdidaktiken diskutiert und dort auch sehr unterschiedlich beantwortet (Blömeke u. Suhl, 2010).

Ein bedeutsamer Aspekt bei Shulmans Modell ist, dass es Wissen beschreibt, das nur teilweise in curricularer Form, d. h. mit dem klaren Ziel exakt dieses Wissen zu vermitteln, gelehrt werden kann. Während dies bei Fachwissen eindeutig ist, ergeben sich bei fachdidaktischem Wissen erste Schwierigkeiten wenn es z. B. um die Einschätzung geeigneter Analogien und Repräsentationsformen geht. Auch wenn vielfach angenommen wird, dass die Fähigkeiten von Lehrern durch persönliche Eigenschaften beeinflusst wird, werden insbesondere kognitive Merkmale, wie fachliches Wissen und die persönliche Überzeugung im Laufe der Ausbildung erworben (vgl. Kunter u. Pohlmann, 2009). Zu welchem Zeitpunkt dies geschieht und welche Prozesse daran beteiligt sind, ist allerdings nicht bekannt.

2.4.2. Umgang von Lehrenden mit Fehlern

Dass heute mit dem Begriff *Fehler* auch positive Konnotationen verbunden sind, geht in besonderem Maße zurück auf die von Oser vor mehr als 20 Jahren postulierte *positive Fehlerkultur* (Oser u. Spychinger, 2005; Reusser, 1999). Ihm und seiner Arbeitsgruppe gelang es wissenschaftlich zu belegen, dass Fehler produktiv genutzt werden können und sie damit ein Bestandteil des Lernprozesses sind. Dies machte es möglich, Fehler nicht mehr als zu vermeidendes Übel anzusehen, sondern ihr zwangsläufiges Auftreten zu nutzen.

Ein Teil dieser Fehlerkultur ist das *negative Wissen* nach Oser u. Spychinger (2005), das eine Wissensstruktur von falschem Wissen darstellt. Sie definieren

vier Typen negativen Wissens, die dieses anhand des Kontextes ihres Auftretens klassifizieren:

- negativ deklaratives Wissen
- negativ prozedurales Wissen
- negativ strategisches Wissen
- negativ schemataorientiertes Wissen

Der Aufbau dieser Wissenskonstrukte geschieht nicht zwingend durch eigene Erfahrungen, sondern kann auch basierend auf Beobachtungen, Erzählungen oder auch Lehrmaterialien erfolgen. Es handelt sich damit um einen grundsätzlich einfachen Lernprozess. Dies ist besonders Lernbereichen unabdingbar, in denen Fehler nicht nachgestellt bzw. simuliert werden können oder diese mit Risiken verbunden sind. So weiß ein Schüler, dass er im Chemieunterricht bei Experimenten eine Schutzbrille tragen sollte, ohne jemals die Folgen erlebt zu haben, die aus einer Nichtbeachtung resultieren könnten.

Das negative Wissen bildet ein Gegenstück zum Wissen und dient als Schutzfunktion für das positive Wissen, da es dieses verdeutlicht und auch zu einer Abgrenzung beitragen kann (Prediger u. Wittmann, 2009). Das Wissen darüber, was etwas nicht ist, ist somit gleichermaßen ein Wissen darüber, was etwas ist (Oser u. Spychinger, 2005). Dies ist kein rein theoretisches Konstrukt, das definiert wie ein Wissenserwerb erfolgt, sondern findet auch im Alltag Anwendung. Beispielsweise ist es bei der Bearbeitung von Multiple-Choice-Aufgaben ein gängiger Ansatz, zunächst eindeutig falsche Antworten auszuschließen, um so die Anzahl der Möglichkeiten zu reduzieren und so zur richtigen Lösung zu gelangen.

Die damit eng verknüpfte Theorie des *Conceptual Change* (deutsch: Konzeptwechsel) ist ein insbesondere in den Naturwissenschaften häufig angewandtes Modell, das den Lernweg beschreibt, um von einer vorhandenen Vorstellung zu einer veränderten Vorstellung zu gelangen. Diese Änderung kann eine Aktualisierung sein, die in der Informatik z. B. durch eine Fortentwicklung von Themengebieten nötig ist, aber auch eine Anknüpfung an Alltagsvorstellungen (Präkonzepte) und die potentielle Korrektur von Fehlern. Der traditionelle Konzeptwechsel ist eine abrupte Änderung eines vorhandenen Konzepts zu einem neuen Konzept, während andere Modelle einen langsamen Übergang darstellen, der abhängig vom jeweiligen Kontext ist (vgl. Tao u. Gunstone, 1999).

Elementar ist, dass der Lernende nicht zwangsläufig lernt, wenn er einen Fehler macht oder diesen erkennt, sondern erst durch die Reflexion. Diese kann nicht

erfolgen, wenn der Lernprozess als Prüfungssituation gestaltet wird und jede Antwort systematisch mit „richtig“ oder „falsch“ bewertet wird (vgl. Hammerer, 2001, S.37-38). Oser u. Spychinger (2005, S. 164) definieren dazu die konträr zueinander stehende Vermeidungsdidaktik und die Fehlerermutigungs- und Fehleraufsuchdidaktik. In ersterer bewertet der Lehrende Fehler negativ oder übergeht diese komplett, während in den letztgenannten Methoden der Lehrende eine bewusste Auswertung und Reflexion der Fehler vornimmt.

Auch wenn verschiedene Metaanalysen zur Wirksamkeit eines Feedbacks zu sehr unterschiedlichen Ergebnissen kamen, so gilt die Wirksamkeit als bewiesen (z. B. Bangert-Drowns u. a., 1991; Mory, 2004). Beispielhaft sei hier die Klassifikation von Feedback nach Jacobs (1998) genannt, die die Qualität und den Nutzen in aufsteigender Reihenfolge repräsentiert:

- Knowledge of Results (KOR) als einfache Rückmeldung *richtig* oder *falsch*
- Knowledge of Correct Result (KCR) als Angabe der richtigen Lösung
- Answer Until Correct (AUC) als Wiederholung von KOR bis der Lernende die richtige Antwort nennt
- Elaborated Feedback als Feedback, das über die zuvor genannten Vorgehensweisen hinaus geht

Eine Reaktion des Lehrenden auf einen Fehler ist nur dann möglich, wenn dieser auch als solcher identifiziert wird (vgl. Seifried u. a., 2010, S. 140). Dies kollidiert allerdings mit der geringen Fehlerhäufigkeit im Klassenunterricht. Heinze (2005) beobachtete im Rahmen einer Videostudie von Mathematikunterricht durchschnittlich fünf Fehler pro Unterrichtsstunde. Dies ist einerseits auf die unterrichtsbedingte Vermeidung von Falschantworten durch den Lernenden zurückzuführen. Andererseits aber auch auf den Lehrer, der eine Unterbrechung des Unterrichts durch einen Fehler vermeiden möchte. Dieser Umstand führt dazu, dass auch den Fehlvorstellungen keine besondere Beachtung geschenkt werden kann, da diese oftmals gar nicht sichtbar werden.

Die Frage, ob eine Fehlvorstellung überhaupt im regulären Unterrichtsverlauf identifizierbar ist, wird noch ergänzt durch die Frage, wie sie weiter analysiert werden kann, um sie einerseits zu ersetzen, andererseits aber auch ein erneutes Auftreten zu verhindern. Reiss u. Hammer (2010) nennen sowohl fachliches als auch fachdidaktisches Wissen als Voraussetzung für eine erfolgreiche Diagnose von Fehlern und Fehlvorstellungen: diese besteht nicht aus einem reinen Erkennen einer Falschantwort, sondern stellt eine weiterführende Analyse dar, um die Logik eines Fehler zu identifizieren und damit die Fehlerursache zu erkennen. Als Beispiel

werden drei Aufgaben zur Bruchrechnung aufgeführt (vgl. Abbildung 2.1), die systematische Fehler zeigen.

Aufgabe

Welche Fehler beim Umgang mit Brüchen vermuten Sie?

1. $\frac{3}{4} + \frac{2}{3} = \frac{5}{7}$
2. $\frac{4}{5} \cdot \frac{3}{5} = \frac{12}{5}$
3. $\frac{3}{4} \cdot 3 = \frac{1}{4}$

Abbildung 2.1.: Aufgabe zur Prüfung fachdidaktischen Wissens über Fehlvorstellungen nach Reiss u. Hammer (2010)

Diese Aufgabe zeigt, dass eine simple Bewertung im Sinne von „falsch gerechnet“ nicht ausreicht. In diesem Fall haben die Rechnungen klare Fehlermuster, die auf eine falsche Anwendung von Rechenregeln hindeuten. Diese müssen somit zunächst identifiziert werden, indem der Lösungsweg analysiert und nachvollzogen wird. Es erscheint naheliegend, dass zu diesem Zweck auch Wissen über typische Fehler und Fehlvorstellungen hilfreich ist. Die zu dieser Aufgabenstellung gehörigen Lösungen sind in Abbildung 2.2 zu finden.

Lösung

1. Übertragung des Schemas „Zähler mal Zähler durch Nenner mal Nenner“ auf die Addition.
2. Übertragung des Schemas der Addition gleichnamiger Brüche auf die Multiplikation.
3. Verwechslung von Multiplikation und Division.

Abbildung 2.2.: Lösungen zur Aufgabe in Abbildung 2.1

2.4.3. Empirische Ergebnisse zum Wissen über Fehlvorstellungen

Die zuvor genannten Forschungsergebnisse repräsentieren vornehmlich theoretische Konstrukte, die nicht oder nur ansatzweise empirisch validiert wurden. Im Folgenden sollen daher einige empirische Ergebnisse zum Wissen bzw. Kompetenzen im Gebiet der Fehlvorstellungen vorgestellt werden.

KCS

Insbesondere im Rahmen empirischer Erhebungen ergab sich die Frage, ob sich die PCK tatsächlich instrumentalisieren lässt, d. h. ob sie sich durch eine Erhebung messen lässt. Hill u. a. haben mit diesem Hintergrund die Teachers' Knowledge of Content and Students (KCS) definiert (vgl. Abbildung 2.3):

> „KCS is used in tasks of teaching that involve attending to both the specific content and something particular about learners, for instance, how students typically learn to add fractions and the mistakes or misconceptions that commonly arise during this process. In teaching students to add fractions, a teacher might be aware that students, who often have difficulty with the multiplicative nature of fractions, are likely to add the numerators and denominators of two fractions." (Hill u. a., 2008)

Die KCS ist als Teil der PCK modelliert, die sich hier ausserdem in die Bereiche Knowledge of Content and Teaching (KCT) und „Knowledge of Curriculum" gliedert. Zu ersterem findet eine Abgrenzung statt, indem diese als „teaching moves" definiert werden, zu denen u. a. Unterrichtskonzepte im Allgemeinen oder das Vermeiden und Verbessern von Schülerfehlern gehören. Die KCS hingegen ist das Wissen über mögliche Fehlvorstellungen als solche. Diese Beschreibung macht deutlich, dass sich beide Bereiche schwer voneinander abgrenzen lassen und eng zusammenhängen.

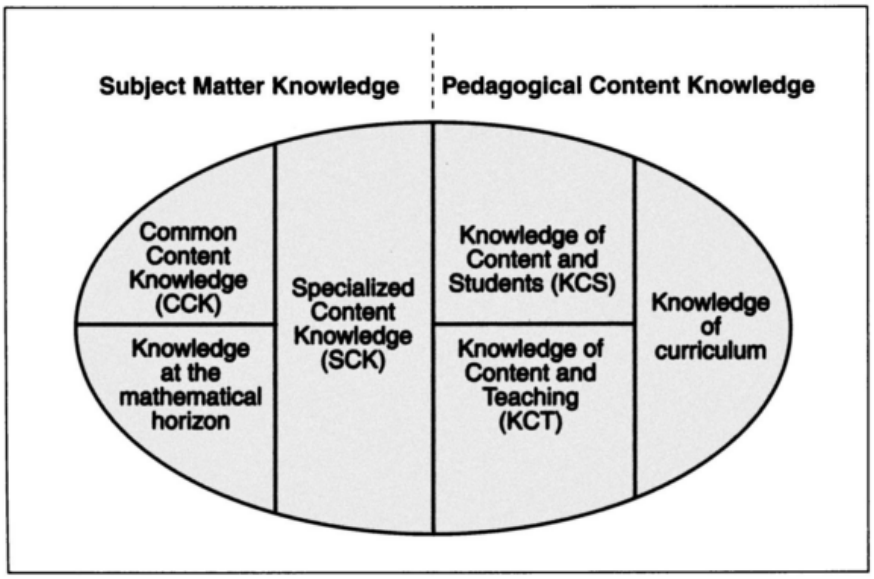


Abbildung 2.3.: Knowledge of Content and Students (KCS) und Knowledge of Content and Teaching (KCT) nach Hill u. a. (2008)

Aufbauend auf dieser Definition wurden im zuvor vorgestellten Forschungsansatz Testitems entwickelt, die in die folgenden Kategorien eingeteilt wurden (Hill u. a., 2008, S. 380):

- Common student errors: identifying and providing explanations for errors, having a sense for what errors arise with what content, etc.
- Students' understanding of content: interpreting student productions as sufficient to show understanding, deciding which student productions indicate better understanding, etc.
- Student developmental sequences: identifying the problem types, topics, or mathematical activities that are easier/more difficult at particular ages, knowing what students typically learn „first", having a sense for what third graders might be able to do, etc.
- Common student computational strategies: being familiar with landmark numbers, fact families, etc.

Die Items wurden in Form von Multiple-Choice-Aufgaben entworfen. Die Eignung des Formats wurde nach Durchführung der Studie hinterfragt, da eine Faktorenanalyse nicht die tatsächliche Messung der KCS belegen konnte. Die Teilnehmer hatten zumeist alle möglichen Antworten angekreuzt, da diese aus ihrer Perspekti-

ve nachvollziehbar schienen. Dementsprechend war es nicht möglich, anhand der Aufgabenlösungen das tatsächliche Wissen abzubilden. Im Rahmen von Interviews, die zu den Items geführt wurden, zeigte sich hingegen deutlich, dass die in der Definition der KCS beschriebenen Kenntnisse zur Beantwortung der Aufgaben genutzt wurden (Hill u. a., 2008, S. 391).

2.4.4. Didaktische Rekonstruktion

Die didaktische Rekonstruktion stellt einen Rahmen mit verschiedenen Forschungsschritten für die Lehr-Lern-Forschung dar, der sich seit dem Ende der 1990er Jahre in den naturwissenschaftlichen Fachdidaktiken etabliert hat. Das Ziel beschreibt Kattmann wie folgt:

> „Die Didaktische Rekonstruktion ist als ein Forschungsrahmen entwickelt worden, der Untersuchungen auf genuin fachdidaktische Fragestellungen hin orientiert. Dies betrifft in erster Linie diejenigen Forschungsvorhaben, die einen fachlich konzeptuellen Bezug haben und sich nicht allein auf allgemeine Unterrichtsprozesse und Lerndispositionen beziehen. Als „Modell“ bildet sie fachliches Lernen und Lehren ab und bezieht sich dabei auf Teiltheorien zum fachlichen Lernen und Lehren, die systematisch zusammengeführt werden.“ (Kattmann, 2007)

Es handelt sich um eine Vorgehensweise, die das Ziel hat, fachwissenschaftliche Inhalte so aufzubereiten, dass diese an das vorhandene Wissen der Lernenden anknüpfen können. Ganz zentral ist, insbesondere im Hinblick auf potentielle Fehlvorstellungen, dass die Rekonstruktion wechselseitig erfolgt. In der fachlichen Klärung (vgl. Abbildung 2.4) wird zunächst die fachwissenschaftliche Perspektive erfasst, die neben dem Wissen auch Methoden, Theorien und Termini umfasst. Gleichzeitig werden durch die Lernpotential-Diagnose die Lernenden analysiert, indem neben vorhandenen Vorstellungen (im Sinne von Präkonzepten) auch kognitive Grenzen und Fertigkeiten betrachtet werden. Diese beiden Schritte werden in der didaktische Strukturierung zusammengebracht und aufeinander bezogen.

Ein Teil der didaktische Rekonstruktion im Bereich der Lernpotential-Diagnose ist es, durch Interviews, Tests oder aber auf Basis von Erfahrungen mögliche Fehlvorstellungen von Lernenden zu identifizieren. Dies bezieht sich nicht nur auf die reine Messung und Ermittlung, sondern auch eine Analyse, wie diese entstanden sind und welche Ursachen zu identifizieren sind. Diese Ergebnisse können dann im nächsten Schritt auf ihre Entwicklung von der Vorstellung zum fachlich richtigen

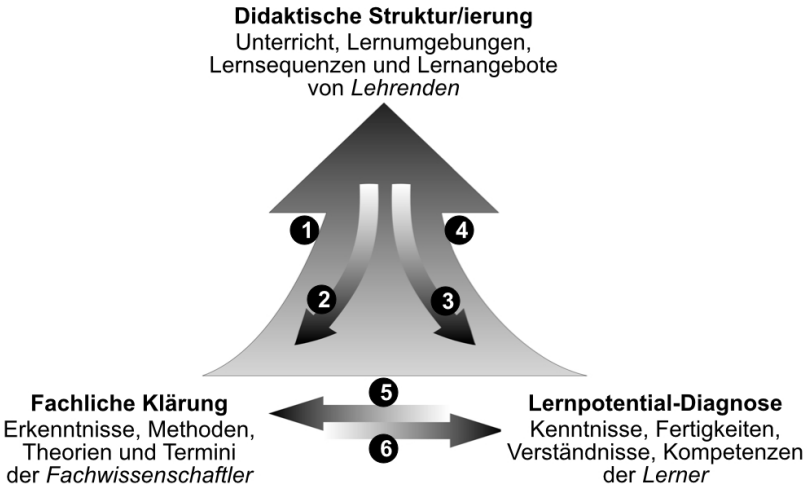


Abbildung 2.4.: Modell der didaktischen Rekonstruktion nach Kattmann (2007)

Wissen untersucht werden. In der didaktischen Rekonstruktion besteht die Annahme, dass das Wissen der Lernenden zunächst als konsistent und richtig wahrgenommen wird, dies jedoch nicht zwangsläufig ist. Es ist daher bedeutsam, die Denkstrukturen soweit zu analysieren und zu hinterfragen, so dass daraus ein schlüssiges Modell entstehen kann.

Diethelm u. a. (2011) erweitern das Modell für den Informatikunterricht, indem sie die Lehrerperspektive ergänzen. Dies wird damit begründet, dass die Lehrerausbildung in Deutschland sehr heterogen ist und damit auch diese ein relevanter Faktor ist. Die Autoren vermuten, dass die Ergebnisse der didaktischen Rekonstruktion durch unterschiedliche Vorbildung und geringen fachlichen Austausch voneinander stark abweichen. Diese Argumentation lässt sich auf andere Bereiche übertragen, in denen Lehrende keine einheitliche Ausbildung durchlaufen und auch generell keine einheitliche Vorbildung haben (insbesondere Dozenten an Universitäten).

An dieser Stelle muss allerdings hinterfragt werden, ob dies tatsächlich ein Umstand ist, der sich aus der heterogenen Ausbildung ergibt und auch spezifisch für das Fach Informatik ist. Es ist offensichtlich, dass auch Lehrende, die eine strukturierte Ausbildung als Informatiklehrer durchlaufen haben, unterschiedliche Kenntnisse der einzelnen Themen haben und diese entsprechend auch in ihren Unterricht

einfließen lassen. Einen empirischen Beleg, dass diese Perspektive tatsächlich starke Auswirkungen hat, gibt es allerdings nicht.

2.4.5. Fachwissen

Aus der vorhergehenden Beschreibung und den präsentierten Modellen wird deutlich, dass das Fachwissen von Lehrenden einen Einfluss auf die Fähigkeit hat, Fehlvorstellungen zuverlässig und erfolgreich zu identifizieren, sowie auch Lehrmaterialien hinsichtlich potentieller Fehlvorstellungen einschätzen zu können. Dies ist zunächst einmal grundsätzlich nachvollziehbar, da es ohne Wissen über das jeweilige gelehrte Konzept unmöglich wäre eine Fehlvorstellung zu erkennen. Wie groß allerdings der Einfluss ist, ist schwierig zu messen. Insbesondere muss die Frage gestellt werden, ob ein größeres Fachwissen eventuell Defizite im Bereich der PCK ausgleichen kann. Einer umfangreichen Messung dieser Zusammenhänge haben sich bisher nur wenige Forschungsprojekte gestellt.

So konnte die Studie Cognitive Activation in the Classroom (COACTIV) (Krauss u. a., 2008a,b) zeigen, dass Fachwissen und fachdidaktisches Wissen bei Gymnasiallehrer so hoch korrelieren, dass sie aus empirischer Sicht nicht zu trennen sind. Gleichzeitig wurde festgestellt, dass die Unterrichtserfahrung (gemessen in Form der Berufstätigkeit in Jahren) nicht mit dem Fachwissen korreliert. So kann davon ausgegangen werden, dass der Wissenserwerb zum Zeitpunkt des Berufeintritts weitgehend abgeschlossen ist.

Die Ergebnisse der Studie legen nahe, dass das reine Fachwissen bei der Entwicklung fachdidaktischen Wissens eine große Rolle spielt. Diese Ergebnisse fügen sich perfekt in die zuvor beschriebenen Theorien zur Entwicklung des Wissens über Fehlvorstellungen ein.

2.5. Schülerkognition

Der Begriff „Schülerkognition“ bezeichnet das Wissen über fachspezifische Fehler von Schülern und typische Schwierigkeiten und schließt damit auch das Wissen über Fehlvorstellungen ein. Er fand zwar in früher angesiedelten wissenschaftlichen Studien Verwendung, wurde aber erst durch das fachdidaktischen Modell von COACTIV spezifiziert und mit einer tiefergehenden Bedeutung gefüllt.

Schon Shulman schloss in seine Definition der PCK auch die Schülerkognition ein:

> „Pedagogical content knowledge also includes an understanding of what makes the learning of specific topics easy or difficult: the conceptions and preconceptions that students of different ages and backgrounds bring with them to the learning of those most frequently taught topics and lessons. If those preconceptions are misconceptions, which they so often are, teachers need knowledge of the strategies most likely to be fruitful in reorganizing the understanding of learners, because those learners are unlikely to appear before them as blank slates. “ (Shulman, 1987, S. 9-10)

Dieser Ansatz wurde in zahlreichen Modellen aufgegriffen und fortgeführt. An (2004) sieht in ihrem mathematikdidaktischen Modell das Unterrichten als zentralen Punkt der PCK, der das Wissen über das Denken von Schülern (*knowing students' thinking*) umfasst und sich in vier weitere Domänen aufgliedert. Darin eingeschlossen ist auch das Wissen über Fehlvorstellungen.

In verschiedenen fachdidaktischen Forschungsprojekten wird die Schülerkognition allerdings auch mit dem Wissen über Fehlvorstellungen gleichgesetzt. Die Autorin der vorliegenden Arbeit sieht darin allerdings, insbesondere im Kontext dieser Arbeit, drei grundlegende Probleme:

- Der Begriff gehört nicht zum häufig genutzten Vokabular der Lernpsychologie und Didaktik. Er ist dementsprechend eher unbekannt und muss erläutert bzw. definiert werden. Diese Definition ist gleichzeitig schwierig, da keine eindeutige, mehrfach in der Literatur zitierte und verwendete existiert.
- Die Schülerkognition umfasst nicht nur Wissen über Fehler, Schwierigkeiten und Vorstellungen von Lernenden, sondern z. B. auch Wissen über die Zusammensetzung der Lerngruppe und ihre Besonderheiten. Fehlvorstellungen sind somit nur eine der Komponenten. Da in dieser Arbeit der Schwerpunkt allerdings genau darauf liegt, könnte dieser Begriff irreführend sein.
- Die Verwendung des Begriffs Schülerkognition als fachdidaktische Kompetenz in Verbindung mit dem Wissen über Schülerfehlern und Fehlvorstellungen impliziert einen Zusammenhang. Dieser kann aus theoretischer Sicht vermutet werden, wurde allerdings nie empirisch bestätigt. Ganz im Gegenteil haben Messungen gezeigt, dass auch Fachwissen ganz entscheidend dazu beiträgt, Fehlvorstellungen zu identifizieren. Dies schließt zwar nicht aus, dass fachdidaktisches Wissen wichtig und wertvoll ist, bzw. die Fähigkeit fördert,

Fehlvorstellungen zu identifizieren. Dieses sollte allerdings nicht vorausgesetzt werden.

Insbesondere der letztgenannte Punkt ist der zentraler Gegenstand dieser Arbeit, da die Messung des Wissens über Fehlvorstellungen das Ziel hat, Zusammenhänge festzustellen und zu ermitteln, welche Faktoren Einfluss haben (vgl. Kapitel 1). Dies durch eine vorausgehende Begriffsdefinition zu beeinflussen und den Zusammenhang praktisch vorzugeben, ist nicht gewünscht.

2.6. Fehlvorstellungen als Thema in der Ausbildung und Lehre

Im Vergleich zu anderen Fächern ist die Anzahl der Standards und der Empfehlungen, die sich mit der fachspezifischen und fachdidaktischen Ausbildung von Informatiklehrkräften beschäftigen (und damit auch mit dem Aspekten der Fehlvorstellungen), gering.

Die schon 1979 veröffentlichten Empfehlungen der Gesellschaft für Informatik e.V. (GI) zur Ausbildung, Fortbildung und Weiterbildung von Lehrkräften in der Informatik gelten als eine der ersten dieser Art. Darin wurden vier inhaltliche Schwerpunkte aufgeführt, von denen einer die Didaktik der Informatik ist. Die aktuellste Auflage gibt Empfehlungen für die universitäre Lehrerausbildung, die in die Ausbildung von Lehrern in der Sekundarstufe I und II, von Lehrern an kaufmännisch-berufsbildenden Schulen, sowie für die informationstechnische Grundbildung aufgeteilt ist. Die folgenden Angaben beziehen sich auf die Ausbildung von Lehrern in der Sekundarstufe I und II. Im Bereich der fachdidaktischen Kompetenzen werden Fähigkeiten genannt, die sich auf die Auswahl und Reflexion geeigneter Unterrichtsinhalte und curricularer Empfehlungen und die lerngruppenadäquate Vermittlung und Methodenbeherrschung beziehen. Fachspezifische Lehr-Lern-Prozesse finden keine Beachtung.

Ein international vielbeachtete Veröffentlichung ist der von Hazzan u. a. (2011) konzipierte Entwurf eines *Methods of Teaching Computer Science (MCTS)* Kurses, der insbesondere für seine praktische Auslegung gelobt wurde. Dieser hat das Ziel, das fachdidaktische Wissen (nach Definition der PCK) zu erweitern:

> „[...] From this perspective, the MTCS course aims at broadening the students' PCK and sets the basis for the in-school training that takes

> place after it. In other words, based on the working assumption that the computer science prospective teachers learn the discipline of computer science and general pedagogy topics in other courses, the MTCS course focuses on the uniqueness of teaching computer science.“(Hazzan u. a., 2011, S. 3)

Der Kurs orientiert sich nicht an einem strengen Ablaufplan, sondern stellt informatikdidaktische Konzepte und Ansätze vor, die mit Hilfe von Übungen vertieft werden. Diese bestehen z. B. aus einem Perspektivwechsel (der Studierende nimmt die Rolle des Schülers ein) oder aus der Zusammenstellen von Überblickswissen. In Kapitel 6 „Learners' Alternative Conceptions“ widmen sich die Autoren der Identifikation von Fehlvorstellungen. Der Begriff „Alternative Conceptions“ wird hier verwendet, um die Zulässigkeit von entsprechenden Konzepten in einer frühen Lernphase zu betonen.

Als besondere Schwierigkeiten bei der Identifikation von Fehlvorstellungen nennen (Hazzan u. a., 2011, S. 80) drei Punkte:

1. Understanding how people do not understand topics which they conceive as trivial ones;
2. Lowering their level of understanding to that of a novice learner since their understanding of the subject area is usually more advanced;
3. „Getting into the head“ of someone else (not just a pupil) because they have not gained yet the experience needed for performing such tasks.

Diese Aufzählung begründen sie mit dem Modell nach Fuller (1973) (zitiert nach Hazzan u. a. (2011)), das drei Entwicklungsstufen eines Lehrers beschreibt, in denen er von der Selbstzentrierung („self stage“) zur Zentrierung auf die Schüler („impact stage“) übergeht. In letzterer Stufe, in denen sich erfahrene Lehrer zumeist befinden, stehen nicht mehr die grundlegenden Verhaltensweisen vor einer Klasse (z. B. eine geplante Stunde umzusetzen oder die Schüler ruhig zu halten) im Mittelpunkt, sondern die Denkweisen und das Verhalten der Schüler. Der Unterricht läuft quasi automatisiert ab, ohne dass grundlegende Faktoren oder Abläufe (wie z. B. die zeitliche Planung) ständig bewusst kontrolliert werden. Auch wenn diese Stufung nachvollziehbar und angemessen ist, so kann das Wissen über Fehlvorstellungen nicht automatisch eingeschlossen werden. Wie zuvor in Unterabschnitt 2.4.5 dargestellt, spielt durch die hohe Korrelation auch das Fachwissen eine bedeutsame Rolle in diesem Kontext. Gleichzeitig wird suggeriert, dass sich auch das Wissen über

Fehlvorstellungen mit der Erfahrung des Lehrenden ausbildet, was in mehreren Studien nicht bestätigt werden konnte (z. B. Krauss u. a., 2008b).

Nichtsdestotrotz stellt das Buch verschiedene Aufgaben vor, die Lehrern in der Ausbildung eine Hilfestellung geben sollen, Antworten der Lernenden zu interpretieren und Fehlvorstellungen zu identifizieren. So werden falsche Aufgabenlösungen und zugehörige, mit Schülern geführte Interviews, präsentiert, auf deren Basis die Fehler erklärt und begründet werden sollen. Insbesondere die praktische Anwendung sticht in diesem Ansatz heraus, da oftmals in fachdidaktischen Publikationen lediglich auf die Existenz und den Ursprung von Fehlvorstellungen eingegangen wird.

2.7. Schlussfolgerung

Zusammenfassend wurde in diesem Kapitel deutlich, dass Fehlvorstellungen ein sehr komplexes Forschungsgebiet darstellen, das zwar in verschiedenen Einzelaspekten theoretisch wie auch empirisch erforscht wurde, aber immer noch schwierig zu erfassen ist. Dies ist in besonderem Maße darin begründet, dass Fehlvorstellungen nach außen „unsichtbar“ sind. Sie sind bei ihrem Entstehen nicht klar erkennbar und verbergen sich auch später, so dass nicht nachvollziehbar ist, wie sie überhaupt entstanden sind. Dies kann zudem dazu führen, dass Lernende Aufgaben zu einem Thema, zu dem sie eine Fehlvorstellung haben, richtig bearbeiten können, aber diese trotzdem „mitnehmen“, so dass sie zu einem späteren Zeitpunkt (z. B. im weiteren Verlauf der Ausbildung oder bei der Übertragung auf einem anderen Kontext) erst wieder von Bedeutung ist. Der Prozess der Identifikation ist häufig aufwändig und meist nur durch Interviews bzw. gezielte Fragestellungen durchführbar. Eine eindeutige und vor allem auch vollständige Diagnose und Entfernung der Fehlvorstellung ist auch ohne diese Gegebenheiten schwierig durchzuführen.

Ein Lehrender muss sich intensiv mit den Vorstellungen der Lernenden beschäftigen, um diese überhaupt zu erkennen und analysieren zu können. Dies ist in der Schule schwierig, in der Universität aufgrund der Größe einer Veranstaltungen nahezu unmöglich. In letzterer wäre dies lediglich in kleineren Lerngruppen wie z. B. Tutorien möglich, allerdings besteht hier das Problem, dass diese meist von Lehrenden durchgeführt werden, die keine oder nur wenige Semester Lehrerfahrung haben. Es ist daher wichtig, präventiv mit Fehlvorstellungen umzugehen. Das bedeutet, dass Beispiele nicht nur anhand zunächst naheliegender Kriterien wie der Alltagsnähe oder auch ganz grundsätzlich nach der Verständlichkeit ausgewählt

werden, sondern auch auf mögliche Missverständnisse, die zur Entwicklung von Fehlvorstellungen führen können, überprüft werden.

Der Lernende muss eine Fehlvorstellung erst als solche identifizieren, bevor er sie benennen und auch beschreiben kann. Dies ist jedoch selbstständig kaum möglich. Viel mehr wird das vorhandene Wissen (d. h. einschließlich der Fehlvorstellung) leicht angepasst, so dass es weiterhin genutzt werden kann. Solche Anpassungen können über mehrere Jahre hinweg fortgeführt werden, ohne dass die zugrundeliegende Fehlvorstellung erkannt wird.

Die Frage, ob schriftliche Beschreibungen bzw. Aufgabenstellungen tatsächlich helfen, das Wissen über Fehlvorstellungen bei Lehrenden zu verbreitern, ist nicht zu beantworten. Sicher ist, dass auf diesem Weg niemals ein vollständiger Katalog erstellt werden kann, der dazu dient eine Diagnose zu erstellen, welche Fehlvorstellung hinter welchem Fehler stehen könnte. Nichtsdestotrotz ist naheliegend, dass durch das Verstehen beispielhafter Fehlvorstellungen zumindest ein Grundwissen herausgebildet wird, das als Basis dienen kann. Alleine das Bewusstsein dafür, wie Fehlvorstellungen entstehen und sich entwickeln können, scheint wertvoll zu sein.

In diesem Kapitel wurden zunächst nur Forschungsergebnisse zu Fehlern und Fehlvorstellungen aus der Lernpsychologie und der allgemeinen Didaktik vorgestellt. Die Informatik wurde darin nur ansatzweise berücksichtigt. Im folgenden Kapitel soll daher eine Analyse der Forschung zu Fehlvorstellungen in der Informatik erfolgen.

3. Fehlvorstellungen in der Informatik: Bewertung und Analyse

3.1. Überblick

Im Folgenden wird auf Basis der zuvor vorgestellten Grundlagen und Definitionen ein Überblick über wissenschaftliche Arbeiten zu Fehlvorstellungen in der Informatik gegeben. Ziel ist es, zunächst eine Zusammenfassung des aktuellen Forschungsstandes zu präsentieren und diesen daraufhin hinsichtlich verschiedener Kriterien zu betrachten und zu strukturieren. Dies umfasst eine Zusammenfassung der Ergebnisse von Erhebungen und der angewandten Methodik, sowie der Ursachen für Fehlvorstellungen. Anschließend erfolgt ein Wechsel in die Praxis der Informatikausbildung, indem analysiert wird, inwieweit die theoretischen Ergebnisse in der Praxis angewandt werden können oder bereits angewandt wurden und inwieweit das Thema in der Ausbildung von Informatiklehrkräften präsent ist. Abgeschlossen wird dies durch eine Zusammenfassung und eine Einordnung in den Kontext der Arbeit.

Abgesehen von einzelnen Beispielen werden an dieser Stelle noch keine Fehlvorstellungen umfassend vorgestellt, sondern erst im Rahmen der Entwicklung des Testinstruments in den Forschungsverlauf dieser Arbeit einfließen (in Kapitel 5). Dieses Kapitel bildet somit eine Brücke von der didaktischen, pädagogischen und psychologischen Forschung zu Fehlvorstellungen im vorhergehenden Kapitel zur Informatik und der damit verbundenen Forschung.

3.2. Einleitung

Grundsätzlich ist festzustellen, dass das Forschungsfeld Fehlvorstellungen in der Informatik schwierig zu erfassen ist. Ein Grund hierfür ist, dass der Begriff Fehlvorstellung (bzw. „misconception“) in vielen relevanten Publikationen nur am Rande erwähnt wird. Insbesondere frühere Veröffentlichungen bezeichnen diese oftmals als „Fehler“ und nehmen im Text eine Abgrenzung zum eigentlichen Fehler vor. Dies macht es schwierig, entsprechende Arbeiten zu identifizieren und damit auch, die Geschichte der Forschung zu beschreiben. Nur durch das konsequente Zurückverfolgen von Zitaten und die Recherche nach verwandten Publikationen ist es an dieser Stelle gelungen, eine umfangreiche Sammlung an Ergebnissen zusammenzutragen.

Es wurden insgesamt mehr als 150 Veröffentlichungen zu Fehlvorstellungen in der Informatik analysiert, die als relevant für das Forschungsgebiet gelten und vornehmlich ab den 1980er Jahren publiziert wurden. Zusätzlich wurden weitere Publikationen aus verwandten Fachdisziplinen herangezogen. Diese lassen sich zwar nicht ohne Einschränkungen auf die Informatik übertragen, dennoch sind sie zur Bewertung und Einschätzung hilfreich, indem sie zum Vergleich inhaltlicher Schwerpunkte, aber auch zum Vergleich des methodischen Vorgehens genutzt werden. Gleichzeitig sind sie von Interesse, da sie insbesondere für frühe Studien in der Informatik eine bedeutsame Grundlage bilden.

Die Analyse der relevanten Forschungsprojekte ergibt eine deutliche Konzentration auf universitäre Lernumgebungen. Schulische Forschungsergebnisse beschränken sich zumeist auf Praxisberichte, die konkrete Konzepte und Vorgehensweisen vorstellen. Häufig erfolgt ein Vergleich unterschiedlicher Lehr- und Lernansätze, der dabei mögliche Fehlvorstellungen implizit oder explizit einschließt. Umfassende empirische Ergebnisse sind in diesem Umfeld jedoch rar und meist an sehr eingeschränkte Ziel- bzw. Messgruppen gebunden. Leider bedeutet dies auch, dass die Größen der Stichproben oft nicht ausreichend sind, um Rückschlüsse auf die Grundgesamtheit zu ziehen.

Erwartungsgemäß betrachtet die Mehrheit der Forschungsergebnisse „Anfänger“ im Gebiet der Informatik (vgl. Unterabschnitt 3.3.2), im universitären Umfeld die Teilnehmer der international als CS1 und CS2 bezeichneten Kurse. Dies ist insbesondere in der Häufigkeit des Auftretens und der Messbarkeit der Fehlvorstellungen begründet, da sich deutlich besser nachvollziehen lässt, wann und in welchem Umfeld Wissen erworben wurde. Dies verschwimmt im Laufe der Ausbildung immer mehr, da zusätzliche Faktoren hinzukommen, so dass sich im Regelfall nicht mehr alle Faktoren einzeln betrachten lassen (vgl. Kapitel 2). So entsteht allein durch die

Belegung von Wahlveranstaltungen oder Kursen mit wechselnder Thematik (z.B. Seminaren) ein sehr unterschiedlicher Wissensstand bei den Studierenden. Auch wenn die Gruppe der Studienanfänger in der Informatik sehr heterogen hinsichtlich ihrer Vorkenntnisse ist, so unterscheidet sie sich zumindest im nationalen Vergleich nicht deutlich.

3.3. Vorstellung und Analyse des aktuellen Forschungsstandes zu Fehlvorstellungen in der Informatik

3.3.1. Historische Einordnung und Entwicklung

Die ersten Veröffentlichungen zum Thema Fehlvorstellungen in der Informatik zu finden ist herausfordernd. Der Grund dafür ist, dass der Begriff im Allgemeinen (d. h. nicht nur in informatischen Publikationen) erst spät definiert und verwendet wurde. Wann dies genau geschah, ist schwer nachzuvollziehen, allerdings findet sich die „Fehlvorstellung“ (bzw. misconception) nur in sehr wenigen der bis zu Beginn der 1990er Jahre veröffentlichten relevanten Forschungsergebnisse. Zumeist beschäftigt man sich simpel mit der Frage, wie Anfänger in der Informatik lernen und welche Verständnisprobleme auftreten können. Dass es dort um Fehlvorstellungen geht, ist erst aus dem Kontext ersichtlich.

Denning veröffentlichte 1975 einen Artikel über Fehlvorstellungen zur strukturierten Programmierung (Denning, 1975). Er bezieht sich auf eine allgemeine Denkweise bzw. auf eine allgemeine Sicht auf die Programmierung, die er nicht mit dem Lernprozess in Zusammenhang bringt. Dies steht zwar nur indirekt in Verbindung mit dem hier verwendeten Begriff der Fehlvorstellungen, ist aber trotzdem erwähnenswert, da es sich dabei um eine der ersten Publikationen in der Informatik handelt, die den Begriff „misconception“ überhaupt verwendet und mit Inhalt füllt.

Weniger Jahre später veröffentlicht Mayer (1979) einen Artikel, in dem er sich damit beschäftigt, welche Konzepte Teilnehmer eines BASIC Kurs lernen. Darauf aufbauend wurde 1981 der Artikel „The Psychology of How Novices Learn Computer Programming“ (Mayer, 1981) veröffentlicht, der eine der meistzitierten Veröffentlichungen im Gebiet der Fehlvorstellungen in der Informatik ist. Er befasst sich mit der langfristigen Vermittlung von Programmierkonzepten und hinterfragt,

wie das Verständnis verbessert werden kann. Der Autor überträgt hierzu Methoden aus der Bildungsforschung und der kognitiven Psychologie auf die Informatik:

> „The goal of this paper is to explore techniques for increasing the novice's understanding of computer programming by exploring techniques that activate the "appropriate anchoring ideas."" (Mayer, 1981, S. 122)

Diese Thematik wird im Folgejahr wieder von Bayman u. Mayer (1982) aufgegriffen, die sich mit der Entwicklung von Vorstellungen beim Erlernen der Programmiersprache BASIC beschäftigen, die nicht zwangsläufig korrekt und nützlich sind. Die Autoren betrachten diese als „mentale Modelle" und messen sie bei dreißig Studierenden, die an einem BASIC-Kurs teilgenommen haben. Es gelang ihnen, mehrere Fehlvorstellungen zu identifizieren, die sich bei der Verwendung der Programmiersprache zeigten (z. B. bei den Befehlen GOTO und PRINT).

Auch Bonar u. Soloway befassten sich mit der Ursache von Fehlvorstellungen:

> „In particular, we focus on the knowledge that novice programmers bring to these early programming problems. Our key idea is that many novice programming bugs can be explained as inappropriate use of the knowledge used in writing step-by-step procedural specifications in natural language." (Bonar u. Soloway, 1985, S. 134)

Sie verwenden hier den Begriff „bug" als einen Fehler in einem Programm, der auf fehlende Kenntnisse des Programmierers zurückzuführen ist und beziehen sich dabei explizit auf diesen. Sie beschreiben weiter, dass fehlende Kenntnisse durch einen „patch" ausgeglichen werden, der häufig nicht richtig ist und so zu dem „bug" führt. Dies spiegelt eine bekannte Ursache für die Entstehung von Fehlvorstellungen wider: fehlendes Wissen führt dazu, dass eine Aufgabe (in diesem Fall eine Programmieraufgabe) nicht gelöst werden kann. Um dies auszugleichen, wird eine falsche Schlussfolgerung gezogen, die zwar offensichtlich zu einer Lösung führt, aber die Problemstellung nicht beantwortet und somit nicht richtig ist. Signifikant für diese Publikation ist, dass eine sehr technische und durch Begriffe aus der Informatik geprägte Sprache verwendet wird. Verweise auf die Bildungswissenschaften oder Didaktik gibt es keine.

Der Zeitrahmen der zuvor genannten frühen Veröffentlichungen zeigt deutlich, dass das Interesse an einer erfolgreichen Vermittlung von informatischen Themen und damit verbunden auch die Vermeidung von Fehlvorstellungen, schon früh entstanden ist. Mit der Einrichtung von Informatik Lehrstühlen an Universitäten in den 60er Jahren begann auch der Entwurf und die Diskussion von Curricula

für die Informatikausbildung (vgl. z. B. Gupta, 2007) und damit gleichzeitig die Beschäftigung mit der Frage, wie Konzepte gelehrt werden und wie die Lernenden mit Themen umgehen sollten.

Ein Großteil der sehr frühen Veröffentlichungen wurde im Gebiet der Human-Computer-Interaction publiziert. Dies ist insbesondere darauf zurückzuführen, dass es bis dato nur vereinzelt Professuren gab, die sich gezielt mit der Didaktik der Informatik beschäftigten. Ab 1970 fanden jährliche Symposien der ACM Special Interest Group in Computer Science Education (SIGCSE) statt, die einen immer größer werdenden Zulauf hatten (vgl. z. B. Tucker, 1996). Parallel dazu scheint das Interesse an der Frage, wie Informatik vermittelt werden kann, stark zuzunehmen. Dies wird auch aus den Veröffentlichungen dieser Zeit deutlich, die genau diese Frage als ihre Forschungsmotivation beschreiben.

Gleichzeitig wächst auch das Spektrum der Forschungsansätze und ihrer Tiefe bis heute. Während zu Beginn der Fokus deutlich auf der Identifizierung und der Beobachtung der Vorstellungen der Lernenden lag, wurden später mehr und mehr Modelle geschaffen und Erkenntnisse aus anderen Disziplinen adaptiert (vgl. z. B. Kaczmarczyk u. a., 2010, S. 108). Ein Beispiel hierfür ist die Herleitung von Ursachen für Fehlvorstellungen. Während in frühen Veröffentlichungen ihre Entwicklung unbeeinflussbar wirkt und das Forschungsziel ihre Identifikation und Beschreibung war, so werden immer mehr Rückschlüsse gezogen, warum Lernende in der Informatik Fehlvorstellungen entwickeln (wie z. B. durch einen falschen Transfer mathematischen Vorwissens). Dies bildet gleichzeitig auch die Verbindung zur „anderen Seite“ der Fehlvorstellungen, nämlich der Frage, wie sie durch einen Lehrenden identifiziert und ersetzt werden können, die wiederum zum Thema dieser Arbeit, der Messung des Wissens über Fehlvorstellungen, führt.

3.3.2. Themengebiete und Zusammenhänge zwischen Themen

Forschungsergebnisse zu Fehlvorstellungen werden nicht nur mit Konzentration auf tatsächliche Fehlvorstellungen veröffentlicht, sondern sind ein „Nebenprodukt“ anderer Studien oder werden als ein Nachteil bzw. Ergebnis von Lehrmethoden und -mitteln gesehen. Auch dies trägt stark zur Unüberschaubarkeit des Forschungsgebietes bei, da hier gemeinhin der Frage nachgegangen wird, wie sich mögliche Fehlvorstellungen vermeiden oder reduzieren lassen und nicht der Frage, wie sie überhaupt entstehen oder zu identifizieren sind. Im Folgenden soll daher ein

möglichst umfassender Überblick darüber gegeben werden, in welchen Forschungskontexten Fehlvorstellungen betrachtet und gemessen werden und wie sie jeweils dort eingebettet und verortet sind.

Programmierung

Wie bereits in Abschnitt 3.1 beschrieben, besteht bei den Publikationen zu Fehlvorstellungen in der Informatik eine große Konzentration auf Programmieranfänger in Bezug auf eine bestimmte Sprache bzw. ein Sprachparadigma. In den 80er Jahren waren dies BASIC (z. B. Bayman u. Mayer, 1982; Bayman u. Mayer, Richard, 1983) und Pascal (z. B. Spohrer u. Soloway, 1986b). Die Veröffentlichungen folgen hier den jeweils aktuellen Entwicklungen in der Praxis (d. h. den in der Lehre eingesetzten Sprachen), was sich auch in den folgenden Jahren fortsetzt.

Pea weist in diesem Kontext aber auch auf Missverständnisse hin, die unabhängig von einer Programmiersprache auftreten:

> „In this article, however, I plan to consider instead the kinds of fundamental and widespread conceptual misunderstandings or „bugs“ in program understanding that appear, from our own and others' work, to be relatively independent of specific commands or programming languages. These misunderstandings, we will argue, have less to do with the design of programming languages than with the problems people have in learning to give instructions to a computer.“ (Pea, 1986, S. 26)

Er geht damit indirekt auf einen zentralen Punkt ein, der bei Fehlvorstellungen von großem Interesse ist: die Ursache für die Entwicklung. Diese kann in der Syntax einer Sprache liegen, aber gleichzeitig auch in grundlegenden Konzepten, die unabhängig von der Sprache sind. So ist ein häufiges Problem bei Programmieranfängern, dass diese dem Computer ein „Mitdenken“ bei der Verarbeitung von Programmcode zusprechen, das dieser nicht kann. Pea beschreibt dies wie folgt:

> „Egocentrism bugs are the flip side of intentionality bugs. Whereas intentionality bugs involve comprehending and tracing what a program will do, egocentrism bugs are involved in creating a program to do something. Each bug type presupposes that the computer can do what it has not been told to do in the program.“ (Pea, 1986, S. 30)

Hier fehlt den Lernenden das Verständnis dafür, dass der Ablauf eines Programms einem strikten Muster folgt und lediglich das verarbeiten oder berechnen kann, was der Programmierer bereitstellt. Auch die iterative Verarbeitung stellt häufig ein Problem dar, weil hier, im Gegensatz zur Kommunikation zwischen Menschen, keine Rücksprünge oder nachgelagerte Erklärungen möglich sind. Eine Aussage im Stil von „... das tust du nur, wenn vorher jenes passiert ist“ ist in einfachen Programmieraufgaben nicht umsetzbar. Das Denkmuster, Anweisungen so zu strukturieren, dass zu jedem Zeitpunkt alle nötigen Informationen verfügbar sind, muss erst erlernt werden (vgl. z. B. Bonar u. Soloway (1985) und Christiaen (1988)).

Seit Beginn der 90er Jahre finden entsprechend der Entwicklung der in der Lehre eingesetzten Sprachparadigmen objektorientierte Sprachen, im speziellen Java, vermehrt Beachtung. Holland u. a. beschreiben verschiedene Fehlvorstellungen, die Studierende über Objekte haben. Sie ziehen dabei speziell den Vergleich zur prozeduralen Programmierung:

> „Many early teaching examples feature classes with a single instance variable. There is a danger that some students with previous experience of procedural programming may generalize prematurely from these examples to develop the misconception that objects are in some sense mere wrappers for variables.“ (Holland u. a., 1997)

In weiteren Studien wurde gezeigt, dass ein Vorwissen in imperativer Programmierung mit dem erfolgreichen Lernen von objektorientierter Programmierung kollidieren kann:

> „Probably because no programming model was presented to the participants, they developed their own models. These are often simple generalizations of previous knowledge obtained during the study of the simpler components of OOP, or (as was probably the case in the population considered in this research) derived from knowledge of imperative programming.“ (Liberman u. a., 2011, S.7)

Gemessen an der wachsenden Anzahl an Publikationen und ihrer Popularität ist die objektorientierte Programmierung das am häufigsten betrachtete Paradigma in der Forschung zu Fehlvorstellungen. Dies bezieht sich insbesondere auf grundlegende Konzepte wie das Anlegen von Klassen.

Dies geht einher mit einer umfangreich geführten Diskussion über den Einsatz von Sprachen und die Nutzung spezifischer pädagogischer und didaktischer Ansätze und Konzepte in der Informatikausbildung generell, insbesondere *objects-first* und

objects-later, *functional-first* und *imperative-first*. Wichtig ist hier, dass die Fehlvorstellungen zwar weiterhin zentrales Thema sind, die Forschungsfrage sich aber auf die Suche eines „optimalen“ Lehrkonzeptes bezieht. Beobachtete und gemessene Fehlvorstellungen werden somit häufig als Indikator verwendet, ob ein Ansatz erfolgreich ist oder nicht. Die beobachteten Fehlvorstellung als solches finden dabei geringe bis keine Beachtung. Stattdessen wird ihre Vermeidung mit Hilfe anderer Ansätze propagiert.

Dass es auf die Fragestellung nach einem optimalen Konzept für das Erlernen von Programmiersprachen keine eindeutige Antwort gibt, zeigt sich in zahlreichen Publikationen. So führten Bennedsen u. Caspersen (2007) eine internationale Erhebung durch, in der sie Lehrende zu den Bestehens- bzw. Durchfallquoten in den Einführungskursen und dem angewandten Konzept befragten. Hier zeigten sich keine erkennbaren Unterschiede. Ehlert u. Schulte (2009) führten eine ähnliche Vergleichsstudie mit Schülern durch. Auch hier liessen sich keine Unterschiede hinsichtlich der Lernergebnisse des objects-first und des objects-later-Ansatzes feststellen. Die im Rahmen entsprechender Studien veröffentlichten Beobachtungen bestätigen aus der Perspektive der Forschung zu Fehlvorstellungen, dass die Auswahl pädagogischer Konzepte und Ansätze (im Sinne von Programmierparadigmen und der Reihenfolge ihrer Vermittlung bzw. der Reihenfolge gelehrter Konzepte) keine direkte und eindeutige Auswirkung auf ihre Herausbildung und Entwicklung hat.

Neben der Betrachtung von pädagogischen bzw. didaktischen Ansätzen, spielt die gelehrte Programmiersprache eine große Rolle in den Forschungsansätzen, da sie als Ursache für die Entwicklung von Fehlvorstellungen gesehen wird. Java hat sich seit dem Einzug der objektorientierten Programmierung in die Lehre zur Standardsprache entwickelt, während die Nutzung von Python, ebenso in der objektorientierten Programmierung, in den letzten Jahren zugenommen hat. In Diskussionen um die Geeignetheit einer Sprache finden sich häufig Fehlvorstellungen als Nachteil wieder. Ein Grund, der als Argument für den Einsatz von Python häufig genannt wird, ist die übersichtliche und einfache Syntax, die vielfach als förderlich für das Erlernen von grundlegenden Konzepten angesehen wird. Es lassen sich im Gegensatz zu anderen Sprachen schon sehr früh Programme erstellen, die von den Lernenden als sinnvoll empfunden werden (Oldham, 2005). Ein entscheidender Unterschied zu Java ist allerdings die dynamische Typverwaltung. Diese führt dazu, dass Lernende zwar die grundsätzliche Typdeklaration und die Nutzung von Variablen ohne Schwierigkeiten erlernen können, aber nicht in der Lage sind, Typen tatsächlich zu unterscheiden (Goldwasser u. Letscher, 2008). Oldham (2005) führt zudem auf, dass die Lernenden die Kompilierung des Python-Codes nicht bewusst wahrnehmen und ihnen somit dieser Schritt im Verständnis fehlt. Alle zuvor

genannten Nachteile stellen gleichzeitig Fehlvorstellungen dar, die unabhängig von dieser Programmiersprache in anderen Veröffentlichungen dokumentiert wurden. Dies ist ein Beispiel dafür, dass Fehlvorstellungen in informatikdidaktischen Studien betrachtet, aber nicht zwingend als solche deklariert werden.

Algorithmen und grundlegende informatische Konzepte

Neben der Programmierung im Speziellen, d. h. der Nutzung konkreter Programmiersprachen und programmiersprachlicher Konzepte, finden in der Forschung zu Fehlvorstellungen auch Themen Beachtung, die unabhängig von Programmiersprachen existieren. Diese Fehlvorstellungen haben gemein, dass sie zwar in Zusammenhang mit dem gelehrten Programmierparadigma stehen, aber unabhängig von diesem gemessen werden können, da sie in einer Pseudosprache dargestellt werden können. Sie lassen sich größtenteils dem Gebiet der Algorithmen zuordnen.

Algorithmen sind ein Themengebiet mit einem großen Alltagsbezug, in dem sich Begriffe und Abläufe, die den Lernenden bekannt sind, wiederfinden. So sind schon junge Schüler damit vertraut, Gegenstände zu sortieren oder zu suchen, ohne dass ihnen bewusst ist, dass sie dabei nach einem Algorithmus vorgehen. Diese Parallele ist insbesondere Einsteigern wenig vertraut, wenn sie Algorithmen zur Problemlösung einsetzen (vgl. z. B. Grimm, 2005). Dementsprechend treten Probleme auf, wenn Kontrollstrukturen gewählt werden müssen, um in natürlicher Sprache formulierte Lösungen oder Abläufe zu implementieren (Sekiya u. Yamaguchi, 2013).

Im Allgemeinen können zu Algorithmen viele Beispiele und Vergleiche genutzt werden, die den Lernenden durch bereits vorhandenes Alltagswissen verständlich sind und dementsprechend nachvollzogen werden können. Die in den Einführungskursen der Informatik gelehrten Sortieralgorithmen können z. B. mit Spielkarten veranschaulicht werden. Dies gilt auch noch für fortgeschrittene Kurse:

> „Metaphors and analogies can be used in all levels of education, starting with the complete basics. For instance, it is well known that a “labeled box” metaphor for a variable in a program helps explain the potentially confusing “x=x+1” statement. But there is no real upper bound on subject difficulty – metaphors are also used in graduate courses at universities.“ (Forišek u. Steinová, 2012)

Dass dies nicht immer erfolgreich ist und die Auswahl von Analogien mit Bedacht erfolgen muss, wird in vielen Publikationen deutlich. So wird für die Rekursion

oftmals das Beispiel der Matroschka Puppen herangezogen (aus Holz gefertigte, hohle Figuren, die sich ineinander schachteln lassen). Sie stehen für die Verschachtelung von Anweisungen und bilden damit auf den ersten Blick eine nachvollziehbare Metapher. Nichtsdestotrotz können hier Fehlvorstellungen beobachtet werden, die vornehmlich daraus entstehen, dass Lernende Details wahrnehmen und der Metapher zuordnen, die nicht Bestandteil dieser sind. Im Fall der Matroschka werden somit der Rekursion Eigenschaften oder Verhaltensweisen zugesprochen, die diese Metapher nicht vermitteln soll. Forišek u. Steinová nennen als eine mögliche Fehlvorstellung die Interpretation, dass jede Anweisung jeweils einen rekursiven Aufruf machen kann: *„With recursion, the main caveat is that each of the dolls always directly contains at most one smaller doll – this sometimes caused false impression that a function can only make one recursive call.“*. Auch Haberman u. Averbuch (2002) erwähnen diese Metapher: *„For example, if the Russian Doll contains a most inner doll that is not decomposable, it might mean that the base case should always be the smallest concrete case of the problem.“*. Dies zeigt, dass sich selbst hinter populären und weit verbreiteten Metaphern Fehlvorstellungen verstecken können.

Ein weiterer didaktischer Ansatz, um Algorithmen zu erklären, sind insbesondere bei jüngeren Lernenden (aber auch bis in universitäre Einführungsveranstaltungen hinein) Rollenspiele. Besonders beliebt sind beispielsweise Simulationen von Sortieralgorithmen, die durchgeführt werden, indem Schüler sich analog zum Ablauf des Algorithmus ihrer Größe nach sortieren. Dies geschieht dadurch, dass sie sich in einer Reihe aufstellen, ihre Größe jeweils mit Größe des Nachbarn vergleichen und bei Bedarf die Plätze tauschen, bis sie mit aufsteigender Größe in einer Reihe stehen. Dies erscheint aus didaktischer Sicht zunächst vielversprechend und reizvoll. Ein erstes Problem ergibt sich allerdings daraus, dass bei diesem Spiel keine „steuernde“ Instanz existiert, die die Elemente vergleicht, sondern die Elemente sich untereinander vergleichen. Dies kann eine erste Hürde bei der Implementierung darstellen. Gleichzeitig ist der Vergleich hier rein optisch, d. h. die Schüler messen oder zählen nicht, sondern betrachten sich gegenseitig und treffen so eine Entscheidung. Entsprechende Faktoren sind meist ohne Schwierigkeiten zu beheben, indem das Rollenspiel leicht umgestaltet wird und damit an die tatsächliche Definition von Algorithmen angepasst wird. Leider sind sie aber oftmals nicht sofort ersichtlich oder bleiben unbeachtet.

Logik

Ein weiteres, besonders früh in der Geschichte der Forschung zu Fehlvorstellungen in der Informatik betrachtetes Thema, ist die Logik. Dieser Umstand ist auf ihre Nähe zur Mathematik zurückzuführen, in der Fehlvorstellungen schon weitaus länger analysiert und dokumentiert werden (vgl. Abschnitt 3.4.2). Nichtsdestotrotz ist die Logik elementarer Bestandteil der Informatikausbildung, auch wenn sie zumeist Inhalt von Mathematikveranstaltungen bzw. -kursen ist. Herman et al. beschreiben diesen Zusammenhang wie folgt:

> „[...] propositional logic and Boolean algebra (hereafter "Boolean logic") play a fundamental role in writing specifications, validating designs, testing rigorously, and optimizing safely. These skills are typically taught early in computer science and computer engineering curricula (typically as part of discrete mathematics or digital logic design classes) and serve as foundations for many of the classes that follow." (Herman u. a., 2008)

Herman u. a. (2008) führten als Grundlage für die Entwicklung eines Content Inventories (vgl. Abschnitt 3.4.1) einstündige „Think-Aloud-Tests" mit Informatikstudierenden durch und stellten ihnen Fragen zu verschiedenen Konzepten der Logik. Die Teilnehmer hatten verschiedene Fehlvorstellungen bei der Anwendung komplizierterer Fälle und reduzierten Aufgabenstellungen auf einfachere, ihnen bekannte, Boolsche Ausdrücke, durch die die Aufgaben nicht gelöst werden konnten.

Wie zuvor im Gebiet der Algorithmen, werden in der Logik Fachtermini genutzt, die den Lernenden bereits aus dem Alltag geläufig sind. Eine Besonderheit ist allerdings, dass Fehlvorstellungen problemlos sichtbar gemacht werden können, wie z.B. durch die Aufstellung falscher Wahrheitstabellen. Konkrete Beispiele für eine falsche Übertragung von natürlicher Sprache in die Logik werden später in Abschnitt 5.6 dieser Arbeit vorgestellt.

Programmierumgebungen

Obwohl die Wahl von Programmierumgebungen nicht direkt mit der Forschung zu Fehlvorstellungen zusammenhängt, sind diese elementarer Bestandteil der Diskussionen um dieses Thema, da sie als bedeutsamer Aspekt für die Entstehung, aber auch für die Vermeidung von Fehlvorstellungen gelten. Im Allgemeinen lässt sich grob zwischen der Nutzung von Pseudocodeumgebungen und auf die Lehre ausgerichtete Programmiersprachen (z. B. Karol oder Logo), graphischen Programmiersprachen

(z. B. Scratch oder Alice), sowie professionellen Programmierumgebungen (z. B. Eclipse oder Netbeans) unterscheiden (nach Jimoyiannis, 2011).

Die Diskussion über Vor- und Nachteile einzelner Programmierumgebungen wird seit Jahren rege geführt. In Fachbüchern zur Informatikdidaktik und Vorlesungsunterlagen werden diese oftmals aufgelistet. Auch auf informatikdidaktischen Konferenzen werden regelmässig neue Ergebnisse zu diesem Thema präsentiert. Bemerkenswert ist dennoch, dass es nur sehr wenige vergleichende Studien gibt, die den Lernprozess über einen repräsentativen Zeitraum hinweg verfolgen und dabei die tatsächlichen Ursachen für Fehlvorstellungen, die in diesem Kontext genannt werden, ermitteln können. Mitunter scheint die Identifikation der Programmierumgebung als Ursache für eine Fehlvorstellung aus einem Schwarz-Weiß-Denken zu resultieren, durch das andere Ursachen konsequent ausgeschlossen werden. Dieser Hintergrund macht es schwierig den entsprechenden Fehlvorstellungen weiteren Kontext und Inhalt zu geben.

So vergleichen Ruf u. a. (2014) den Lernerfolg hinsichtlich bestimmter informatischer Konzepte und die Motivation von Schülern, die Scratch und „Karel the Robot“ in einer Unterrichtsreihe im Umfang von 18 Stunden genutzt haben. Obwohl dieser Ansatz vielversprechend ist, zeigen sich in der Methodik und in der Aussagekraft Mängel. Zum einen wurden die Lernergebnisse in den beiden Kursen mit unterschiedlichen Tests mit jeweils zwei Aufgaben gemessen, die an die Programmierumgebung angepasst wurden. Auch wenn die Schwierigkeit als gleich eingeschätzt wurde, muss die Frage nach der Validität der Ergebnisse gestellt werden. Zum anderen gilt dies für den Test als solches, da mit zwei Programmieraufgaben lediglich ein Bruchteil der nicht nur langfristig zu vermittelnden Kompetenzen gemessen werden kann. Dies erschwert es zu vergleichen, inwieweit Konzepte besser oder schlechter mit einer bestimmten Programmierumgebung gelernt werden können oder ob mit einer Programmierumgebung bestimmte Fehlvorstellungen verursacht und provoziert werden.

Gleichzeitig sollte generell vermieden werden, voreilige Schlussfolgerungen zu ziehen, durch die als Ursache für eine Fehlvorstellung eine Programmierumgebung identifiziert wird. Die Ursachen für Fehlvorstellungen können vielfältige Ursachen haben, die sich häufig nur individuell nachvollziehen lassen. Eine einmalige Messung, die zeigt, dass eine spezifische Fehlvorstellung bei Nutzern einer Umgebung vorhanden ist, bei Nutzern einer anderen Umgebung jedoch nicht, ist dafür definitiv zu wenig. Leider schließen viele Studien mit genau diesem Ergebnis ab und lassen offen, welcher Aspekt der Programmierumgebung tatsächlich die Entwicklung von Fehlvorstellungen begünstigt. Im Kontext dieser Arbeit ist dies ausreichend und

gewinnbringend, da auf diesem Weg dokumentierte Fehlvorstellungen bereitgestellt werden. Die Schlussfolgerungen, die daraus gezogen werden, sollten aber nicht unreflektiert übernommen werden.

Hardware

Der Großteil der Hardwarekomponenten eines Computers ist aus Perspektive der Lernenden grundsätzlich greifbar und aus dem Alltag bekannt. Begriffe wie „Prozessor“, „Festplatte“ oder „Arbeitsspeicher“ sind Nutzern durch Beschreibungen von Anwenderprodukten bekannt. Ihre grundlegende Funktion aus Perspektive des Anwenders ist damit geläufig. Die spezifische Funktionsweise und der Aufbau einzelner Komponenten ist hingegen oft unbekannt.

Ioannou u. Angeli erläutern die besondere Rolle und Schwierigkeit von Hardwarekomponenten eines Computers bei der Thematisierung in der Lehre am Beispiel der CPU:

> „The CPU is a topic difficult to be conceived by students, due to teachers' difficulty to effectively represent and teach how the CPU operates. Students have no knowledge of electronics and therefore cannot explain the way it operates.“ (Ioannou u. Angeli, 2014)

Das Problem besteht hier insbesondere darin, die Funktionsweise so zu reduzieren und simplifizieren, dass sie für die Lernenden einerseits verständlich ist, aber auch ausreichend, um das erworbene Wissen z. B. bei der Programmierung nutzen zu können. Dass dieses Wissen dabei als elementar angesehen werden kann, zeigen hardwareorientierte Ansätze in der Lehre. So schlägt das ACM/IEEE Curriculum von 2001 neben *imperative-first*, *objects-first*, *functional-first*, *breadth-first* und *algorithms-first* auch *hardware-first* für Einführungsveranstaltungen vor:

> „The hardware-first approach teaches the basics of computer science beginning at the machine level and building up toward more abstract concepts. The basic philosophy behind this strategy is for students to learn about computing in a step-by-step fashion that requires as little mystification as possible.“ (The Joint Task Force on Computing Curricula, 2001)

Diese Herangehensweise ist zweifellos nicht sehr verbreitet und stellt eher einen Ansatz in einer elektrotechnischen Ausbildung dar. Er drängt zwangsläufig die informatischen Inhalte in den Hintergrund und kann daher demotivierend für die

Lernenden sein. Nichtsdestotrotz bestätigt der Ansatz die Relevanz des Wissens über Hardware in Einführungsveranstaltungen. Wie zuvor in Kapitel 2 beschrieben, besteht auch hier das Risiko, dass Funktionsweisen, die beobachtet werden (in diesem Fall bei der Programmierung) und Vorwissen falsch verknüpft werden, sowie falsche Schlussfolgerungen aus Beobachtungen gezogen werden.

Ben-Ari u. Kolikant (1999); Ben-Ari (2005), sowie Yehezkel u. a. (2001) vertreten in mehreren Publikationen die These, dass das erfolgreiche Erlernen einer Programmiersprache nicht nur an das praktische Wissen über die Anwendung der Programmiersprache geknüpft ist, sondern an das Modell der Sprache als solches, z. B. ihre Speicherverwaltung. Wenn hier Fehlvorstellungen vorhanden sind, haben diese Einfluß auf die Programmierung. Dies wird von Rountree u. a. (2013) unterstützt, die empirisch zeigen, dass ein grundlegendes Wissen über Hardware wichtig ist, um Polymorphismus und Vererbung in Java zu verstehen und nutzen zu können.

Nutzererfahrung

Die Rolle des Vorwissens von Lernenden wurde zuvor schon mehrfach dargestellt. Diese stellt damit ein eigenes Forschungsfeld dar, das das Ziel hat, Nutzererfahrungen und -verhalten zu erheben und zu interpretieren. Dies bezieht sich insbesondere darauf, Erfahrungen und Wissen über die Informatik zu ermitteln, das ausserhalb der Ausbildung gewonnen wurde. Dieses Ziel ist eng verknüpft mit der Motivation, das Wissen von Studierenden zu Beginn ihres Studiums zu messen (vgl. Abschnitt 3.3.3). Dies kann auf einem eher geringen Niveau geschehen, um Erfahrungen mit der Nutzung von Computern oder dem Internet zu messen, aber auch in Form eines fortgeschrittenen Tests, in dem Anfänger nach ihrem intuitiven Vorgehen bei der Formulierung von Algorithmen gefragt werden.

Hammond u. Rogers (2006) beschäftigten sich mit dem Wissen von Kindern über Computer und entsprechende Konzepte, insbesondere, operationale Prozesse. In einer Pilotstudie wurde das Wissen während der Nutzung gemessen, was sich allerdings durch die fehlende Aufmerksamkeit der Probanden als nicht erfolgreich erwies. Die Interviews wurden daraufhin unabhängig geführt und anschließend unter Zuhilfenahme verschiedener Wissensstufen analysiert. Diese reichen von der einfachen Vertrautheit mit Konzepten, über die Anwendung, bis hin zu einer vertieften Kenntnis von Funktionsweisen und Zusammenhängen. Diese Stufung ist eng verknüpft mit der Wissenstaxonomie nach Anderson u. Krathwohl (2001). Ein Ergebnis der Studie ist, dass das Wissen der Kinder zwar mit der Erfahrung wächst,

ihre Erklärungen für bestimmte Konzepte aber aus wissenschaftlicher Perspektive nicht besser werden. Sie tragen dabei häufig einmal erschlossene Erklärungen weiter, ohne diese zu verändern oder selbst zu hinterfragen. Dies ist der Art und Weise sehr ähnlich, wie Fehlvorstellungen einen Lernprozess überdauern können (vgl. Kapitel 2) und bestätigt, dass der Lehrende hier intervenieren muss, indem er das zunächst offensichtlich vorhandene Wissen der Lernenden korrekt einschätzt und es gegebenenfalls korrigiert oder erweitert.

Auch Diethelm u. a. (2012) befragten Internetnutzer im Alter von 13 und 14 Jahren (Klasse 7 und 8) zu ihren Vorstellungen über Übertragungstechniken, den Aufbau des Internets und diversen Internetdiensten (Chat, Instant Messenger, u. a.). Die Interviews wurden jeweils in Zweiergruppen mit fünf Frageblöcken von jeweils fünf oder zehn Minuten Länge durchgeführt. Zusätzlich wurden Materialien wie Lego Steine oder Papier und Stifte zur Förderung der Kreativität zur Verfügung gestellt. Auf diesem Weg wurden verschiedene mentale Modelle zu grundlegenden Konzepten ermittelt, die bei der Entwicklung von Testitems in Kapitel 5 noch einmal näher betrachtet werden sollen.

3.3.3. Motivation der Forschungsvorhaben zu Fehlvorstellungen in der Informatik

Die Motivation der Forschungsvorhaben bzw. -ergebnisse, in denen Fehlvorstellungen in der Informatik relevant sind, lässt sich im Allgemeinen in zwei Kategorien einordnen: zum einen eine analysierende oder vergleichende Leistungsbewertung der Lernenden. Dazu gehört zumeist eine Betrachtung verschiedener Lerngruppen, die einen Vergleich von Lerngruppen (z. B. Studierende im Hauptfach Informatik und Ingenieure wie Tew u. a. (2005)) oder Lehrkonzepte (z. B. Objects-First und Objects-Later in der objektorientierten Programmierung wie z. B. Ehlert u. Schulte (2009); Sanders u. Thomas (2007)) anstrebt. Insbesondere auf ersteres Ziel wird im folgenden Abschnitt weiter eingegangen. Zum anderen kann als zweite Kategorie die Förderung von Lernprozessen und die Konzeption von Lehr- und Lernkonzepten angesehen werden. Die Motivation für diese Forschungsansätze sind zumeist Fehleinschätzungen der Lernergebnisse durch Lehrende, die entweder am Ende eines Kurses (d. h. im Rahmen einer Prüfung) oder aber in der weiterführenden Ausbildung sichtbar werden. Hier entsteht die Frage, warum Lernende nicht über das Wissen und die Kompetenzen verfügen, die sie mit Abschluss einer vorhergehenden Lerneinheiten hätten erlangen sollen. Ähnlich wie bei zuvor genannten Studien

sind Fehlvorstellungen in diesem Fall mehr ein Nebenergebnis, das als Indikator verwendet wird.

Die Granularität der Forschungsansätze wird, wie schon im vorherigen Abschnitt dargestellt, im Laufe der Jahre deutlich feiner. Während in den ersten Jahren noch „Programming“ das zentrale Forschungsgebiet war (z. B. Denning, 1975), veränderte sich dies immer weiter zu einzelnen Kursen und Programmiersprachen (z. B. Bayman u. Mayer, 1982) bis hin zu einzelnen Bestandteilen von Programmiersprachen wie z. B. Datenstrukturen (Paul u. Vahrenhold, 2013). Dementsprechend kann auch die Motivation deutlich klarer beschrieben werden.

Der grundlegende Ansatz scheint sich ebenso im Laufe der Jahre zu wandeln. So leiten Spohrer u. Soloway (1986a) ihren Forschungsansatz mit „programming bugs“ ein, die sie bei Programmieranfängern beobachtet haben. Die Motivation ist sehr auf das praktische Ergebnisse ausgerichtet, nicht auf die Lernenden selbst (vgl. Lang u. Beauboeuf, 2012). Ein Grund hierfür ist vermutlich der Forschungshintergrund, da die Mehrheit der Studien in diesen Jahren nicht der Informatikdidaktik entstammt, sondern der allgemeinen Informatik bzw. insbesondere der Human-Computer-Interaction. Sie konzentriert sich stark auf die Beobachtung der Lernenden und ihren Umgang mit informatischen Themen. In den Folgejahren lässt sich deutlich erkennen, dass immer mehr pädagogische und didaktische Aspekte in die Forschung eingebracht werden. Gleichzeitig werden einzelne Lehransätze hinterfragt oder zumindest mit dem Verlauf des Lernprozesses in Verbindung gebracht.

Messung der Wissens von Anfängern

Das Wissen von Schülern und Studierenden zu Beginn der Informatikausbildung ist wie bereits erwähnt sehr heterogen. Dies ist bei ersteren durch Erfahrungen ausserhalb der Schule bedingt, bei letzteren zusätzlich durch eine sehr unterschiedliche Ausbildung in der Schule. Diese Situation stellt in der Praxis eine Herausforderung dar, weil die unterschiedlichen Leistungsstufen in den gemeinsamen Unterricht integriert werden müssen, um keine dieser Stufen auszuschließen. Eine große Schwierigkeit besteht aber auch darin, zunächst adäquat das Vorwissen und das Verständnis für grundlegende Konzepte der Informatik bei den Lernenden zu beurteilen und einzuschätzen. Analog zu den in Kapitel 2 beschrieben Risiken, besteht die Gefahr, das Fehlen von konkretem fachlichen Wissen mit nicht vorhandenem Wissen gleichzusetzen.

Bonar u. Soloway betrachteten diese Situation in ihrer bereits mehrfach erwähnten Studie. Sie nennen hier als Faktoren Alltagswissen, das die Lernenden bereits in Einführungskurse in die Programmierung mitbringen:

> „Novices bring a knowledge of standard tasks in step-by-step natural language procedures to their introductory programming course. Examples of such tasks include looping, making choices, and specifying sequences of actions.“ (Bonar u. Soloway, 1985, S. 137)

Auch Simon u. a. greifen dies in ihrer Studie zum Vorwissen von Studierenden vor ihrer ersten Informatikvorlesung auf und beschreiben ihre Motivation wie folgt:

> „This disconnect between demonstrated programming knowledge and demonstrated reasoning skills suggests students have considerable knowledge that we, as instructors, can leverage to teach computer science more effectively. This idea lies squarely within the constructivist tradition, which holds that students learn by building on what they already know. To leverage students' existing knowledge, however, we must first determine what that knowledge is.“ (Simon u. a., 2006)

Sie beziehen sich hier somit implizit auf den Übergang zwischen dem unstrukturierten Vorwissen und dem darauf folgenden strukturierten Unterricht und messen hierzu die Präkonzepte (englisch: preconceptions), die bei den Lernenden vorhanden sind. In der Erhebung zeigte sich, dass 69% der Anfänger ohne Programmierkenntnisse einen Algorithmus beschreiben konnten, der zehn Zahlen aufsteigend nach ihrer Größe sortiert. Auch wenn ihre Darstellung häufig aus Sicht eines Programmierers nicht effektiv war, zeigt dies dennoch, dass hier schon Fähigkeiten im Problemlösen vorhanden sind, auf die in der Einführungsvorlesung aufgebaut werden kann. Dies schließt insbesondere die Verwendung von alltagsnahen Kontexten ein (Lewandowski u. a., 2007).

Bemerkenswert ist, dass in der betreffenden Studie die Ergebnisse nach 10 Wochen schlechter waren, da insbesondere Fachterminologie nicht korrekt in natürlicher Sprache beschrieben werden konnten und mehr Fehlvorstellungen gemessen wurden. Dies ist ein Indiz dafür, dass hier ein Prozess stattfindet, durch den das informelle Vorwissen durch das Fachwissen ersetzt wird, dieses aber gleichzeitig noch zu fragil ist, um bereits selbstständig beschrieben und erläutert zu werden. Die Lernenden versuchen somit ihr neues Wissen einzusetzen, beherrschen dies allerdings nicht auf einer Ebene, auf der sie dies auch in einer argumentativen und nicht anwendungsbezogenen Form tun könnten. Ähnliche Studien wurden auch in späteren Veröffentlichungen durchgeführt und kamen zu vergleichbaren Ergebnissen.

Die ersten Wochen nach Beginn der Ausbildung sind daher hinsichtlich der Entwicklung von Fehlvorstellungen prägend, da die Verknüpfung mit zuvor erworbenen Alltagswissen problematisch sein kann (vgl. Kapitel 2). Gleichzeitig wird hier ein Grundgerüst gebildet, auf das später Erlerntes aufbaut. Fehlvorstellungen sollten somit so schnell wie möglich identifiziert und ersetzt werden.

Entwicklung von Instrumenten

Eng mit der vorhergehenden Motivation verknüpft ist die konkrete Entwicklung von Testinstrumenten, insbesondere Content Inventories (zur weiteren Erläuterung und Erklärung vgl. Abschnitt 3.4.1). Mit ihrer Hilfe kann am Ende einer Lernphase festgestellt werden, ob das geplante zu erwerbende Wissen tatsächlich erworben wurde. Auch wenn dies zumindest in Universitäten grundsätzlich durch Prüfungen geschieht, sind diese letztendlich kein Maßstab für die tatsächliche Qualität bzw. Tiefe des Wissen. Idealerweise findet die Überprüfung zudem außerhalb von Leistungskontrollen statt und beinhaltet eine differenzierte Rückmeldung an die Teilnehmer.

Um dies sicherzustellen, müssen neben der Zusammenstellung von Themen und Inhalten vor allem Distraktoren generiert werden, die real beobachteten Fehlvorstellungen entsprechen und diese zuverlässig messen. Dass dies ein kompliziertes Vorhaben ist, zeigt sich in vielen der veröffentlichten Ansätze, die in zahlreichen Iterationen überarbeitet werden mussten, um zu einem zufriedenstellenden Ergebnis zu gelangen.

3.4. Analyse der Grundlagen, Konzepte und Methoden in der Forschung zu Fehlvorstellungen in der Informatik

Im folgenden sollen verschiedene Herangehensweisen beschrieben werden, die bei der Forschung zu Fehlvorstellungen genutzt werden. Dies bezieht sich zum einen auf konkrete Instrumente und Messmethoden. Zum anderen betrifft dies aber auch die Verwendung von Begriffen und Definitionen und die Einordnungen von Fehlvorstellungen in einen Forschungskontext.

3.4.1. Instrumente und Messungen

Concept Inventories

Ein Schwerpunkt der Identifikation von Fehlvorstellung in vielen Disziplinen ist die Zusammenstellung eines Concept Inventories (CI). Herman definiert die Funktion und das Ziel wie folgt:

> „A CI is a short multiple-choice test that can classify the examinee as someone who thinks in accordance with accepted conceptions in a discipline or in accordance with common misconceptions.“ (Herman, 2011, S. 102).

Mit einem CI ist es somit möglich, den Lernprozess im Anschluss an eine Lehreinheit (z. B. eine Unterrichtseinheit, aber auch eine komplette Vorlesungsreihe) zu überprüfen. Es findet jedoch kein vollständiger Test der Lerninhalte statt, sondern lediglich eine Auswahl der schwierigsten bzw. kritischsten Aspekte. Die idealisierte Grundannahme ist, dass die Lernenden die Inhalte verstanden haben, wenn sie die Fragen des CI richtig beantworten konnten. Im Gegensatz zu einer Prüfung werden keine ausführlichen Aufgaben gestellt, sondern Denkweisen bzw. das Verständnis überprüft. Taylor u. a. (2014) schlagen hier für die Informatik z. B. eine Code Analyse oder Testen von vorgegebenem Code vor. Dies hat zum einen den Vorteil, dass Wissen sehr feingranular abgefragt werden kann, da jeder Aspekt in jeweils einem Testitem überprüft wird. Gleichzeitig ist die Auswertung komplikationslos, da durch die Multiple-Choice Items eindeutige Antworten vorgegeben sind, die nicht zunächst interpretiert und bewertet werden müssen. Dies sichert auch die Vergleichbarkeit zu, wenn Tests an verschiedenen Standorten oder auch von verschiedenen Personen durchgeführt werden.

Der Begriff des CI geht zurück auf das „Force Concept Inventory“ (FCI), das von Hestenes u. a. (1992) vorgestellt wurde. Der FCI-Test besteht aus Multiple-Choice-Fragen zur newtonschen Mechanik, bei denen sich die Schüler bzw. Studierenden zwischen der fachlich richtigen Antwort oder falschen Alltagsvorstellungen entscheiden müssen. Der Test ist mittlerweile zu einem standardisierten Instrument in Universitäten und Schulen in den USA geworden und wurde auch mehrfach in Deutschland eingesetzt (Wilhelm, 2005). Der Einfluss des FCI auf die Lehre wird allgemein als hoch eingeschätzt und er dient auch heute noch als Ausgangspunkt für die Entwicklung weiterer CI (vgl. z. B. Rowe u. Smaill, 2008).

Auch wenn der Aufbau und Inhalt eines CI zumeist simpel erscheint, ist die Entwicklung deutlich aufwändiger als das Formulieren einzelner Testaufgaben und bedarf einer umfangreichen mehrstufigen Validierung. Treagust (1988) empfiehlt hierzu einen 10-stufigen Entwicklungsprozess, der unstrukturierte Schülerinterviews als Grundlage nutzt, um zunächst ein Fehlverständnis bzw. Fehlvorstellungen zu identifizieren und diese insbesondere für Multiple-Choice-Fragen zu nutzen. Adams u. Wieman (2011) weisen jedoch darauf hin, dass lediglich zwei von 16 analysierten Instrumenten Interviews zur Entwicklung und Validierung eingesetzt haben. Sie empfehlen zusätzlich Experten (insbesondere Lehrende) einzusetzen, da diese wertvolle Beiträge zur sprachlichen Umsetzung der Fragen und Antworten leisten können. Allerdings stellte sich heraus, dass auch Testfragen, die von Experten als „perfekt“ eingeschätzt wurden, von den Schülern falsch bzw. abweichend interpretiert wurden (Adams u. Wieman, 2011, S. 1301).

Taylor u. a. (2014) summieren, dass durch vorhandene CI in der Informatik flächendeckend vorhandene Fehlvorstellungen aufgedeckt werden konnten und damit verbunden auch der Einfluss auf die Fähigkeiten Code zu verfolgen, zu lesen und zu schreiben gemessen wurde. Die Ergebnisse zeigten auch, ähnlich wie in anderen Disziplinen, dass Lernende deutlich weniger Wissen haben, als die Lehrenden erwarten. Gleichzeitig machen sie auch auf den Misstand aufmerksam, dass neue Lehrmethoden und Inhalte schnell entwickelt und im Unterricht eingesetzt werden, aber immer noch standardisierte Messinstrumente fehlen, um zu messen, welchen Einfluss diese haben (Taylor u. a., 2014, S. 262).

Almstrum u. a. (2006) entwickelten ein CI für die diskrete Mathematik (DMCI) als Komponente von Curricula für die Informatik. Sie nennen als Grund für die Wahl dieses Themengebietes zum einen die steigende Relevanz, zum anderen aber auch die Rolle und Konstellation des Themas als solches. So lassen sich die Inhalte eines Kurses sehr gut eingrenzen, um sie in einem entsprechenden Instrument zu prüfen. Des Weiteren wurden Schwierigkeiten und Fehlvorstellungen in der diskreten Mathematik schon umfangreich erforscht, was als sehr hilfreiche Quelle für die Konzeption von Aufgaben genutzt werden kann. Sie propagieren damit gleichzeitig ihren umfangreichen Prozess zur Erstellung eines CI. Dieser beginnt bei der Identifikation fundamentaler Konzepte und führt zur Entwicklung von Items mit einer offenen Fragestellung, die anschließend in einer Pilotstudie mit Studierenden erprobt werden. An diesem Punkt liegen somit ausformulierte Antworten der Lernenden vor. Über verschiede Analyse- und Validierungszyklen führt dieser Prozess letztendlich zum fertigen CI. Dieses Vorgehen ist grundsätzlich ähnlich zu den Vorgehen anderer CI-Entwürfe. Ein alternativer Prozess kann verfolgt werden,

indem im Verlauf der Zyklen zusätzlich Feedback von Experten erhoben eingebracht wird und darauf aufbauend die Testitems konzipiert werden.

Herman (2011) haben das Digital Logic Concept Inventory (DLCI) entwickelt, das sich mit Fehlvorstellungen in der digitalen Logik befasst. Die Inhalte wurden mit Hilfe eines mehrstufigen Befragungsverfahrens identifiziert und bewertet. Das erstellte CI wurde mit 440 Studierenden der Universität Illinois getestet. Zudem wurden Experten zu ihrer Einschätzung der einzelnen Items befragt, die überwiegend positiv ausfiel. Besonders interessant ist hier, dass durch das Instrument auch vorhandene Fehlvorstellungen bei Studierenden gefunden werden konnten, die den zuvor besuchten Kurs mit einem guten Ergebnis abgeschlossen hatten. Dies bestätigt somit die im vorhergehenden Kapitel vorgestellte These, dass Fehlvorstellungen nicht notwendigerweise akut die Leistung oder das Wissen beeinflussen.

Ein CI zu Algorithmen und Datenstrukturen wurde von Danielsiek u. a. (2012) erstellt. Sie folgen dabei beiden von Almstrum u. a. (2006) vorgeschlagenen Wegen: der „Feedback-Schleife“, bei der die Eignung der Items regelmässig überprüft wird und entsprechende Anpassungen und Änderungen vorgenommen werden, sowie der Konzeption von Items aufbauend auf Think-Aloud-Interviews mit Studierenden. Dies bestehen daraus, dass der Interviewte eine Aufgabe bekommt, die er im Interview bearbeitet und währenddessen seinen Denkprozess erläutert. Ein interessantes Ergebnis aus dieser Studie ist, dass beide Ansätze zu unterschiedlichen Ergebnissen führen, die sich aber gegenseitig ergänzen. Während durch die Interviews neue Fehlvorstellungen identifiziert werden können, kann durch die Items eindeutig gemessen werden, welche Konzepte von den Studierenden verstanden bzw. nicht verstanden werden. Die beiden Vorgehensweisen profitieren somit voneinander und schließen sich nicht aus, wenn sie parallel angewandt werden, da Ergebnisse in beide Richtungen verifiziert und validiert werden können.

Karpierz u. Wolfman (2014) haben ein CI für vier Fehlvorstellungen entwickelt, die sie im Bereich der binären Suchbäume und Hash Tables identifiziert haben. Auch sie folgen den Empfehlungen von Almstrum u. a. (2006). Dazu haben sie Interviews mit Lehrenden geführt, Prüfungen und Projekte ausgewertet, sowie Think-Aloud-Interviews mit Studierenden geführt. Dieser Ansatz scheint vielversprechend, ist aber bislang nur an einer Universität eingesetzt worden. Auch Webb u. Taylor (2014) entwickelten ein CI, das fokussiert ist auf das Thema Betriebssysteme. Sie verfolgen dabei den Ansatz von Open Source Software, indem das Instrument von verschiedenen Autoren frei entwickelt wurde. Sie sehen hier den Vorteil, dass sich die Erfahrung vieler positiv auf die Qualität auswirkt. Als eines der wenigen CIs ist dieses auch frei verfügbar, jedoch nicht sehr umfangreich und auch wenig erprobt.

Interviews

Interviews können ein Teil der Entwicklung eines CI sein, dienen aber auch in zahlreichen Studien als eigenständige Methoden, um Fehlvorstellungen zu identifizieren und zu analysieren. Sie werden im Regelfall als Think-Aloud-Interviews durchgeführt. Dies hat gegenüber einer „klassischen“ Aufgabenbearbeitung den Vorteil, dass sich diese im Ganzen nachvollziehen lässt. Fehler, die währenddessen auftreten und eventuell im nächsten Schritt korrigiert werden, können trotzdem beobachtet und festgehalten werden. Gleichzeitig besteht die Möglichkeit, bei unvollständigen oder unverständlichen Antworten weitere Fragen zu stellen und dadurch zusätzliche Informationen zu erhalten. Dies bedeutet allerdings auch, dass Interviews so geführt werden müssen, dass sie den Interviewten nicht beeinflussen oder zu Antworten hinführen, die selbstständig nicht getroffen worden wären. Herman u. a. (2012) geben hier explizit an, dass die Interviewten zuvor informiert wurden, dass sie keine Rückmeldung bekommen, ob ihre Antworten richtig sind, aber mehrfach gebeten werden können, ihre Antwort zu vertiefen und zu erweitern. Nichtsdestotrotz stellt ein solches Interview eine andere Situation für den Interviewten dar, da er sich, ähnlich wie in mündlichen Prüfungen, davor scheuen kann eine falsche Antwort zu geben oder nicht verschiedene Ansätze ausprobieren will, die ihn ansonsten zu einer Lösung führen würden. Dies spiegelt sich darin wider, dass die Anzahl der nicht vorhandenen Antworten (d. h. der Lernende sagt, dass er eine Aufgabe nicht bearbeiten kann), im Regelfall bei Interviews höher ist als bei klassischen Tests (Bortz u. Döring, 2006).

Beispiele für nicht optimal geführte Interviews finden sich leider auch im Gebiet der Forschung zu Fehlvorstellungen. Teif u. Hazzan (2006) führten eine Studie zu grundlegenden Konzepten der objektorientierten Programmierung mit Schülern an einer High School durch. In den Interviews, die ausschnittsweise veröffentlicht wurden, ist mehrmals eine Zusammenfassung bzw. Interpretation durch den Interviewer zu finden. Ein Beispiel hierfür ist die folgende Sequenz, die von Teif u. Hazzan (2006, S. 59) veröffentlicht wurde:

> „**Noa**: *I drew a cat here. And it is a class, because there are many many many things in it... and this specific eye* (points with her finger to the encircled eye on her drawing) *is an object.*
>
> **Interviewer**: *In other words, what you are saying is that eye is an object in the class cat, aren't you?*“

Die Antwort des Interviewers macht deutlich, dass hier eine Interpretation durch ihn stattfindet, da Noa vorab nicht den Zusammenhang zwischen Objekt und Klasse erwähnt hat. Eine solche Zusammenfassung und Rückfrage kann durch den Interviewten als eine Bestätigung aufgefasst werden und dadurch intuitiv mit einer Zustimmung beantwortet werden. Dieses Risiko besteht insbesondere dann, wenn im Vorfeld bestimmte Theorien (hier: Fehlvorstellungen) bestätigt bzw. widerlegt werden sollen und der Interviewer daher den Interviewten zu dieser hinführt. Dies ist insofern nachvollziehbar, da Interviews zeitintensiv sind und somit verwendbare Ergebnisse wertvoll sind. Allerdings wird werden valide Ergebnisse bei einer zu deutlichen Hilfestellung wiederum gefährdet.

Dieses Beispiel zeigt, dass Interviews einerseits interessante Ergebnisse liefern können, als einzige und nicht weiter validierte Quelle jedoch problematisch sein können. Wie sich dies umgehen lässt, zeigt sich z. B. in der Methodik, die von Kolikant (2001) verfolgt wird. Er testete zunächst das Wissen von Studierenden, führte sofort danach aber auch Interviews durch, um herauszufinden, wie Antworten oder auch Fehler in den Antworten entstanden sind. Dies hat den Vorteil, dass zunächst der tatsächliche Wissenstest störungsfrei und unter realen Bedingungen ablaufen kann. Diese getrennte Erhebung kann allerdings zur Folge haben, dass der Testteilnehmer eine falsche oder unvollständige Erinnerung hat. Durch einen verzögerungsfreien Ablauf kann dieses Risiko aber minimiert werden.

Code-Reviews

Code-Reviews stellen eine weitere Möglichkeit dar, die Vorstellungen und das Verständnis von Lernenden von Programmcode und Programmiertechniken zu überprüfen, ohne dass diese aktiv und selbstständig programmieren müssen. Dies hat insbesondere bei Programmieranfängern den Vorteil, dass diese nicht auf Kenntnisse von Syntax und Semantik angewiesen sind, um die Aufgabe erfolgreich zu bewältigen. Gleichzeitig sind sie praktisch gezwungen, ihre Kenntnisse natürlichsprachlich zu formulieren, ohne dabei die Möglichkeit zu haben, auf bekannte, auswendig gelernte Codefragmente zurückgreifen zu können. Es findet somit eine Konzentration auf das tatsächliche Verständnis statt.

Turner u. a. (2008) nutzen diese Methodik in einem CS2-Kurs zur objektorientierten Programmierung. Dort sollten Studierende Lösungen zu einer Aufgabenstellung anhand von sechs Kriterien bewerten, die ausführlicher beschrieben wurden, sowie ihre Antwort kurz kommentieren. Die Lösungen, jeweils eine gute und eine schlechte,

wurden nicht speziell für diese Studie generiert, sondern stammten von Studierenden aus dem vorhergehenden Semester. Dabei wurden einige bemerkenswerte Fehleinschätzungen der Testteilnehmer deutlich, die z. B. Code als „gut“ bewerteten, aber als Kommentar dazu Gründe angaben, die exakt für das Gegenteil sprachen, also eine Fehlvorstellung repräsentieren. Auffallend war, dass selbst Studierende, die in vorhergehenden Projekten sehr gut abgeschnitten hatten, beachtliche Fehlvorstellungen zu Themenkomplexen wie beispielsweise der Dekomposition hatten. Dies ist ein weiteres Beispiel dafür, dass ein gutes Abschneiden in regulären Prüfungen nicht bedeutet, dass der Prüfling keine Fehlvorstellungen hat und die Prüfungsinhalte gut beherrscht. Hier scheint das zuvor beschriebene Phänomen aufzutreten, dass Lernende die Anwendung von Wissen beherrschen (hier z. B. lauffähigen Code zu schreiben, der die Aufgabenstellung erfüllt), aber nicht in der Lage sind dieses zu erklären und zu reflektieren.

3.4.2. Verwendung und Definition von Begriffen

Fehlvorstellungen - misconceptions

Zwar existiert in der Literatur keine einheitliche Definition des Begriffs „Fehlvorstellung“, jedoch müssen Fehlvorstellungen stets klar von Fehlern und Verständnishürden getrennt werden (vgl. Kapitel 2). Dass eine Fehlvorstellung eines systematischen Auftretens bedarf, wurde zuvor bereits erläutert. Das impliziert jedoch im Umkehrschluss nicht, dass eine beobachtete Systematik eine Fehlvorstellung bestätigt, d. h. dass ein wiederholt auftretender Fehler, der auch bei verschiedenen Aufgabenstellungen zu beobachten ist, keine Fehlvorstellung sein muss. Insbesondere im Einführungsunterricht bzw. in universitären Einführungsvorlesungen werden grundlegende Konzepte gelehrt, die völlig neu für die Lernenden sind und eine Hürde im Lernprozess darstellen. Es ist daher unvermeidbar, dass diese einer besonderen Beachtung durch den Lehrenden bedürfen. Nichtsdestotrotz handelt es sich in diesem Sinne nicht zwangsläufig um Fehlvorstellungen, da eine hohe Schwierigkeit (häufig als Lernhürde oder Verständnishürde bezeichnet) kein falsches Verständnis impliziert. Diese Unterscheidung ist in vielen veröffentlichten Studien leider nicht eindeutig zu erkennen. So fehlt auch in hier betrachteten Veröffentlichungen mehrfach eine Definition des Begriffs der Fehlvorstellungen (bzw. „misconceptions“) oder alternativ wird ein Auftreten von Fehlvorstellungen beschrieben, bei denen es sich gemäß der in dieser Arbeit formulierten Definition (vgl. Kapitel 2) nicht um Fehlvorstellungen handelt. Beides macht es schwierig, Studien einzuordnen und eine Vergleichbarkeit abzuschätzen.

Nichtsdestotrotz sollen an dieser Stelle einige Definitionen und Beschreibungen zitiert werden, die in Publikationen zu Fehlvorstellungen in der Informatik getroffen wurden. So nutzen Herman u. a. (2012) eine sehr allgemeine und offene Definition (s. auch Herman u. a., 2010; Herman, 2011):

> **Cue**: A perceptual or conceptual trigger that causes a person to retrieve knowledge.
> **Misconception**: The misapplication or mis-cueing of conceptual knowledge within a specific context.

Diese Definition legt nahe, dass falsches bzw. mit falschem Ziel angewandtes konzeptuelles Wissen grundsätzlich einer „misconception“ entspricht. Die Durchführung bzw. die Ergebnisse der Studie zeigen allerdings, dass es hier nicht um die Anwendung konzeptuellen Wissens, sondern um das generelle Vorhandensein dieses Wissens geht. Allerdings sind genau dies Eigenschaften, die eine Fehlvorstellung von einem Fehler unterscheiden. Das schließt ein, dass eine Fehlvorstellung niemals ausschließlich auf Anwendungsszenarien bezogen sein kann und stützt damit auch die zuvor vorgestellten Ergebnisse, in denen Fehlvorstellungen trotz einer korrekten Anwendung des Wissens vorhandenen waren (z. B. von Turner u. a., 2008).

Sudol u. Jaspan formulieren hingegen eine Definition, die der in Kapitel 2 dieser Arbeit formulierten weitgehend entspricht. Sie sehen Fehlvorstellungen als einen dritten Status, zusätzlich zu „richtig“ und „falsch“:

> „The adage that students 'know it or they don't‘ suggestions that either students 'have‘ a particular piece of knowledge, or they have no conception of the topic. Research in the education sciences suggests that there is perhaps a third state, misconception.“ (Sudol u. Jaspan, 2010, S. 32)

Ioannou u. Angeli nutzen in ihrer Definition den Begriff des „alternativen Konzepts“, der ebenso nahelegt, dass eine Fehlvorstellung abseits des klassischen Bewertungsschemas liegen kann. Sie vermeiden damit eine Beurteilung, ob dieses Konzept richtig oder falsch ist, sondern definieren es nur durch den Unterschied vom bewährten oder vermittelten Konzept:

> „The term 'misconceptions‘ is used to describe alternative conceptions, such as theories or views, which are not consistent with concepts currently accepted by the scientific community.“ (Ioannou u. Angeli, 2014, S. 93)

Smith u. a. greifen in ihrer Beschreibung den Begriff „mistake“ auf, der oftmals mit Fehlvorstellungen unpräzise verknüpft wird:

> „Misconceptions have generally been seen as mistakes that impede learning, a view that is difficult to square with the premise that students construct their mathematical and scientific knowledge. Further advances in understanding learning will depend on rethinking the role that students' conceptions of mathematical and scientific phenomena play in sophisticated, expert-like knowledge. Where misconceptions research has focused on the discontinuities between novice students and experts, we will identify and emphasize important dimensions of continuity between them.“ (Smith u. a., 1993)

Aus diesem Zitat wird nicht deutlich, inwieweit sich die Autoren auch auf diese Definition stützen bzw. inwieweit sie diese kritisieren. Nichtsdestotrotz wird auch hier ein Fokus auf die Unterschiede und Zusammenhänge von Anfänger- und Expertenwissen gelegt.

Hinzu kommt, dass es im Allgemeinen schwierig ist, zwischen Fehlvorstellungen und Fehlern zu unterscheiden. So ist es auch nachvollziehbar, dass z. B. Hristova u. a. (2003, S. 153) wiederkehrende Fehler, die sie durch eine Befragung von Lehrern gesammelt haben, als „errors and misconceptions“ bezeichnen. Diese Zusammenfassung ist zutreffend, da zahlreiche Syntax-Fehler beschrieben werden, die zwar wiederholt auftreten, bzw. von mehreren Lehrern genannt wurden, aber sicherlich nicht auf einem grundlegenden Verständnis basieren. Ein Beispiel hierfür ist die Verwendung falscher Klammern im Programmcode. Dies bestätigt einmal mehr, dass auch im weiteren Verlauf dieses Forschungsvorhabens streng darauf geachtet werden sollte, dass dokumentierte Fehlvorstellungen nicht unreflektiert übernommen oder sogar in das Instrument integriert werden, ohne diese auf die Erfüllung der Kriterien für Fehlvorstellungen zu überprüfen, die in dieser Arbeit aufgestellt wurden.

In einigen Fällen wirkt allerdings auch die Argumentation in sich widersprüchlich. So beschreiben Bonar u. Soloway (1985) die von ihnen analysierten Fehler zunächst simpel als fehlende Kenntnisse, betreiben aber anschließend eine ausführliche Fehleranalyse, in der sie insbesondere auf Zusammenhänge zwischen der natürlichen Sprache und der Programmiersprache eingehen und so aus Fehlern Fehlvorstellungen generieren. Dies ist sicherlich nicht falsch und bedeutet nicht, dass diese Fehlvorstellungen nicht existieren. Viele wurden auch in späteren Veröffentlichungen immer wieder dokumentiert. Allerdings sollte hier Vorsicht geboten sein, um einen

Fehler nicht überzuinterpretieren und eine Fehlvorstellung auf ihn zu projizieren, die in dieser Form nicht existiert.

Threshold Concepts

Threshold Concepts (TC) sind ein weiteres Forschungsfeld, die eng mit Fehlvorstellungen verknüpft sind. Meyer und Land definieren sie als „[...] akin to a portal, opening up a new and previously inaccessible way of thinking about something." (Meyer u. Land, 2003, S.1). Sie definieren damit Konzepte, die ein Fach aus Perspektive der Lernenden besonders schwer machen.

In der Literatur existieren zahlreiche Definitionen von Kriterien für TCs, die vornehmlich auf diese fünf Kriterien zurückgehen, die von Meyer u. Land formuliert wurden:

1. Transformative, in that, once understood, its potential effect on student learning and behaviour, is to occasion a significant shift in the perception of a subject, or part thereof.
2. Probably irreversible, in that the change of perspective occasioned by acquisition of a threshold concept is unlikely to be forgotten, or will be unlearned only by considerable effort.
3. Integrative; that is, it exposes the previously hidden interrelatedness of something. Note that if we re-examine the earlier example of opportunity cost from the novice perspective we may observe that while it satisfies 1. and 2. above, it may not be integrative.
4. Possibly often (though not necessarily always) bounded in that any conceptual space will have terminal frontiers, bordering with thresholds into new conceptual areas.
5. Potentially (and possibly inherently) troublesome, for the reasons discussed below.
   (zitiert nach Meyer u. Land, 2003)

Fehlvorstellungen und Threshold Concepts haben gemein, dass sie integrativ sein können, d. h. sie werden im Kontext des strukturierten Wissenserwerbs entwickelt oder aufgenommen. Im Gegensatz zu den Threshold Concepts müssen Fehlvorstellungen das allerdings nicht. Ebenso gemeinsam ist beiden, dass es häufig schwer ist, sie zu vergessen. Lernende entwickeln in diesem Kontext eigene Regeln und Ideen, um ihr Teilwissen einzuordnen und generieren dabei komplexe Zusammenhänge,

die nur schwerlich wieder „gelöscht“ werden können (wie bereits in Kapitel 2 für Fehlvorstellungen erläutert).

Trotz der Gemeinsamkeiten sind Fehlvorstellungen nicht mit Threshold Concepts gleichzusetzen. Dies ist zunächst einmal aufgrund ihrer Definition eindeutig: während Threshold Concepts Konzepte sind, die erlernt werden sollen, sollten Fehlvorstellungen vermieden werden. Durch die dahinterstehende Thematik, auf die sie sich beziehen (z. B. die Rekursion), sind sie aber eng verknüpft. Aus dieser Perspektive erfolgt auch der folgende Vergleich.

Ein wichtiger Unterschied ist zudem, dass die hinter Fehlvorstellungen stehenden Konzepte keine grundlegenden Konzepte sein müssen, die Lernende beherrschen sollten. Sie können in allen Bereichen und mit sehr unterschiedlicher Relevanz entwickelt werden. Darunter können zweifellos auch grundlegende Konzepte sein und gerade darin wird die Relevanz einer Identifikation von Fehlvorstellungen deutlich. Eine ihrer Eigenschaften ist aber auch, dass sie über eine lange Zeit existieren und auch einen langen Lernprozess „überleben“ können, ohne dabei entdeckt zu werden. Dies ist in den meisten Fällen nur möglich, wenn sie nicht so zentral positioniert sind, dass sie immer wieder zu Tage treten, z. B. bei der Bearbeitung von Aufgaben.

Des Weiteren ist zu vermuten, dass Fehlvorstellungen einfacher zu lernen sind als Threshold Concepts (Eckerdal u. a., 2006, S.105). Dies ist insbesondere darin begründet, dass für Fehlvorstellungen nicht der oben genannte Punkt 5 gilt. Fehlvorstellungen entstehen besonders häufig unterbewusst und sind damit nicht Teil eines systematischen Lernprozesses, so dass die Mühsamkeit in diesem Fall fehlt.

Im Allgemeinen lässt sich aber schlussfolgern, dass ein Threshold Concept ein Konzept repräsentiert, zu denen häufig Fehlvorstellungen existieren. Wenn eine Fehlvorstellung für ein Konzept existiert, ist dieses allerdings nicht zwangsläufig ein Threshold Concept. Demnach ist die Forschung über Threshold Concepts auch für diese Arbeit von großer Relevanz.

Diese Relevanz wird auch durch die verschiedenen Punkte deutlich, die Rountree u. a. (2013) als besonders interessant und herausfordernd bei TC bezeichnen:

- Promotes collegiality. To be successful, the approach requires collective responsibility and discourse in order to describe what it means for a learner to “think more like a practitioner”.
- Promotes an approach to teaching and learning that is explicitly disciplinary-situated. It is a response against a “learning outcomes”, managerial approach to monitoring achievement.

- Reminds us that good quality teaching and learning is a transactional inquiry.
- Proposes that learners encounter pivotal points (TCs) that once grasped will transform their thinking in ways that align with our community of practice.
- Embraces unsureness (liminality) as a positive aspect of the learning process.
- Creates a pedagogical partnership that is neither teacher-centred, nor student-centred (Cousin, 2010).
- Encourages qualitative inquiry that opens opportunities for the use of established research techniques from social science in computing education research.
- Provides a means of considering how the professional identity of the student develops – given the transformative nature of thresholds.

Trotz konkreter Kriterien sind Threshold Concepts nicht leicht zu identifizieren oder problemlos zu definieren. Insbesondere ihre Funktion, als ein umfassender Katalog oder eine Sammlung zu gelten, die das Fach strukturiert, macht es schwierig, sie in zuverlässiger und allgemeingültiger Form zu erheben. Gleichzeitig stellt auch der Umgang mit Threshold Concepts in der Lehre eine große Herausforderung dar. Meyer u. Land (2003) beschreiben die Idee der Threshold Concepts selbst als Threshold Concept. Dass der Umgang und die Identifikation schwierig ist, wird auch in der Informatik durch verschiedene unvollständige bzw. sogar gescheiterte Projekte zu diesem Thema belegt (vgl. Abschnitt 4.2.1).

Sanders u. a. (2012) benennen noch ein weiteres Problem bei der Identifikation von Threshold Concepts. Auch wenn ein Konzept als solches keine Herausforderung ist, kann dies durchaus für die Anwendung von diesem gelten. Sie nennen dies „Threshold Skills“. Sie führten dazu 16 Interviews mit Studierenden in Großbritannien, Schweden und der USA durch. In semi-strukturierten Interviews wurden diese nach Konzepten befragt, die sie zu Beginn ihrer Ausbildung zunächst als schwierig empfunden haben. Die transkribierten Interviews wurden anschließend anhand der oben genannten Kriterien für Threshold Concepts und unter besonderer Beachtung der damit verbundenen Fähigkeiten analysiert. Dies ergab, dass Studierende oft das Verständnis eines Konzepts mit der Anwendung gleichsetzen. Sie verstehen zwar das Prinzip, können es aber nicht in einer spezifischen Situation umsetzen. Daraus schlussfolgern die Autoren, dass es nicht ausreichend ist, ein Konzept der Programmierung zu erlernen, indem die Wirkungsweise verstanden wird, sondern dieses auch angewandt werden kann. Ein Beispiel hierfür ist, dass der Lernende nicht nur das Prinzip der Rekursion verstehen muss, sondern auch rekursive Algorithmen programmieren können sollte. Dieses Ergebnis ist allerdings mit Blick auf die „cognitive process dimensions“ nach Anderson u. Krathwohl (2001) (vgl. Abbildung 3.1) nicht erstaunlich. Auch hier bildet „Apply“ (im Sinne

einer einfachen Anwendung) die dritte Stufe einer sechsstufigen Hierarchie und setzt sich damit von „Remember“ und „Understand“ ab.

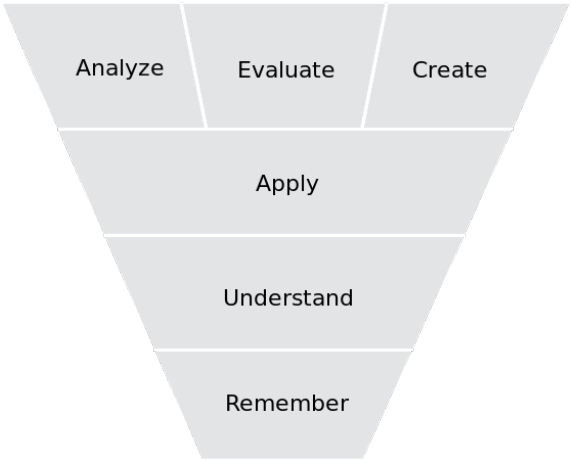


Abbildung 3.1.: Kognitive Prozess-Dimensionen nach (Anderson u. Krathwohl, 2001)

Eckerdal u. a. (2006) verfolgen das Ziel, Threshold Concepts mit verwandten Forschungsergebnissen in der Informatik in Verbindung zu setzen. Interessant im Kontext dieser Arbeit ist die Verbindung zu Fehlvorstellungen und Fundamentalen Ideen. Die Autoren sehen eine große Überlappung der Fundamentalen Ideen und der Threshold Concepts. Beide sind integrativ und sollten von jedem ausgebildeten Informatiker verstanden werden. Dies bedeutet insbesondere, dass sie keine Liste mit wichtigen Themen darstellen, sondern sich bereits in Form einzelner Ideen und Konzepte in diesen wiederfinden. Ein wichtiger Punkt, der Threshold Concepts und fundamentale Ideen (vgl. Abschnitt 4.2.1) unterscheidet, ist allerdings die Vermittelbarkeit. Durch das Vertikalkriterium ist die Bedingung einer fundamentalen Idee, dass sie auf jedem intellektuellen Level vermittelt werden kann. Dies gilt nicht für Threshold Concepts, die ganz im Gegensatz dazu häufig eine besondere Herausforderung darstellen. Sie gelten damit als „boundary markers“:

> „[. . .] they indicate the limits of a conceptual area or the discipline itself. Students who have mastered these Threshold Concepts have, at least in part, crossed over from being outsiders to belonging to the field they are studying.“ (Eckerdal u. a., 2006, S. 103)

Sorva (2010) knüpft an den Vergleich der Fundamentalen Ideen und Threshold Concepts an und erweitert diesen. Er beschreibt „Transliminal Concepts“ als Konzepte, die hinter diesen stehen und die Lernenden zu diesem „locken“. Dies hängt insbesondere damit zusammen, dass sie motivieren können und zudem klare Ziele definiert werden. Sie sind zwar nicht wesentlich für das Verständnis des Konzeptes, können aber einen wichtigen Bestandteil darstellen. Sorva (2010, S. 6) beschreibt sie zudem als weniger generisch und konkreter als das zugehörige Threshold Concept. Als Beispiel nennt er das Java-Interface *construct* für *information hiding* und *polymorphism* für *object interaction*.

Mentale Modelle

Der Begriff der mentalen Modelle („mental models“) findet sich schon in frühen Forschungsergebnissen wieder. Sie repräsentieren ein individuelles Modell, das durch Lernen gebildet und dynamisch angepasst wird und für Entscheidungen und Erklärungen genutzt wird.

Seel definiert sie wie folgt:

> „Mentale Modelle sind als konkrete (...) Repräsentationen unübersichtlicher oder abstrakter Sachverhalte zu begreifen. Sie haben die Funktion, die Objekt- und Ereigniswelt rational verfügbar zu machen und subjektive Plausibilität zu erzeugen.“ (Seel, 2003, S. 258)

In der Forschung zu Fehlvorstellungen in der Informatik finden sie sich beispielsweise in der Form wieder, dass sie für programmiersprachliche Konzepte entwickelt werden:

> „The present study explores the idea that learning of BASIC involves more than the acquisition of specific facts, rules, and skills. Beginning programmers also develop mental models for the language in the process of learning the essentials of BASIC. Users' models, however, may not be accurate or useful ones.“ (Bayman u. Mayer, 1982, S. 5)

Entscheidend ist hier, dass es keine vorgegebenen oder sogar aufbereiteten Inhalte sind, die durch einen Experten an die Lernenden weitergegeben werden. Vielmehr ist es ein individuelles Konstrukt, das dynamisch während des Lernprozesses entsteht. Im Gegensatz dazu ist ein konzeptuelles Model ein vorgegebenes Konzept, das eine angemessene Repräsentation des Lerninhalts darstellt (Wu u. a., 1998). Dies

können z. B. Analogien und Metaphern sein, aber auch abstrakte Modelle wie die mathematische Induktion.

Mentale Modelle können somit hilfreich sein, um Fehlvorstellungen von Lernenden zu identifizieren und gegebenenfalls Ursachen dafür in der Lehre zu finden. Eine Fehlvorstellung ist demnach auch ein mentales Modell, das unvollständig oder generell falsch ist und dementsprechend korrigiert werden muss (Vandegrift u. a., 2010). Es ist jedoch nicht unproblematisch, mentale Modelle abzufragen bzw. diese ausformulieren zu lassen, da sie meist nur als vage Konstrukte existieren und dadurch unbewusst verändert oder angepasst werden, wenn sie erläutert werden müssen (Dicheva u. Close, 2003). Dies liegt in der Natur eines mentalen Modells, da es nur ein abstraktes Denkmodell darstellt, kein explizites Wissen.

Ein mentales Modell führt, ähnlich wie eine Fehlvorstellung, nicht zwingend zu Fehlern in der Anwendung. Das bedeutet, dass beispielsweise Aufgaben, in denen auf ein fehlerhaftes mentales Modell zurückgegriffen werden muss, nicht grundsätzlich falsch bearbeitet werden. Gleichermaßen ist ein valides Modell keine Garantie für die korrekte Anwendung. So beschreiben Vandegrift u. a. (2010) einen Studierenden, der einerseits die Boolsche Gleichung für „if-and-only-if“ aufschreiben konnte, aber nicht in der Lage war, diese in einer praktischen Aufgabe anzuwenden. Dies ist ein Hinweis darauf, dass hier lediglich das explizite Wissen aufgenommen, aber kein mentales Modell ausgebildet wurde, mit dessen Hilfe dieses angewandt werden kann. Zwar konnte kein Zusammenhang zwischen konsistenten mentalen Modellen und den Ergebnissen in CS1 Kursen gefunden werden, jedoch bedeutet dies im Umkehrschluss nicht, dass sie unwichtig order irrelevant sind.

Dicheva u. Close (2003) beschäftigen sich mit mentalen Modellen zur Rekursion und entwickeln eine Methode, um diese mit Hilfe einer Analyse und Codierung von Prüfungsaufgaben zu identifizieren und in ein Kategoriensystem einzuordnen. Sie betonen dabei, dass dies nicht nur den Nutzen hat, Fehlvorstellungen zu identifizieren, sondern auch Lehrenden die Möglichkeit gibt, einen tieferen Einblick in die Wirkung ihres Unterrichts zu erhalten. Zudem könnten Testergebnisse genutzt werden, um Lernende gezielt zu unterrichten. Dicheva u. Close schlagen hierzu die Durchführung von Tutorien vor, die den Fokus jeweils auf das diagnostizierte mentale Modell legen und damit zusammenhängende Fehlvorstellungen korrigieren. Sie geben allerdings auch zu bedenken, dass die Entwicklung mentaler Modelle sehr individuell ist und ein solches strukturiertes Vorgehen niemals alle Schwierigkeiten erfassen kann.

3.4.3. Ausrichtung und Auswahl der Messgruppen

Die Mehrheit der Testergebnisse zu Fehlvorstellungen in der Informatik bezieht sich auf eine sehr eingeschränkte Testgruppe, im Regelfall Teilnehmer einer einzelnen Vorlesung oder einer Vorlesungsreihe. Auch wenn sich durch die Messung verschiedener Generationen dadurch häufig eine große Zahl an Testteilnehmern (teilweise mehr als 1000) ergibt, besteht hier das Problem einer zugrundeliegenden einheitlichen Ausbildung, die die Testergebnisse beeinflussen kann und dadurch auch die Frage nach ihrer Validität aufwirft. Insbesondere muss hier genau beobachtet werden, inwieweit das angewandte Lehrkonzept die Herausbildung von Fehlvorstellungen fördern kann oder diese im schlimmsten Fall sogar produziert. Letzteres ist besonders problematisch, wenn dieser Umstand nicht als Ursache identifiziert wird.

Zudem muss die grundlegende Frage gestellt werden, ob in den Studien und Erhebungen tatsächlich die Fehlvorstellungen der Teilnehmer in der jeweiligen Disziplin gemessen werden oder die Qualität der individuellen Lehre (vgl. u. a. Tew u. Guzdial, 2010). Die Frage der Validität einer Messung stellt sich in vielen Disziplinen. Hierbei ist dies allerdings besonders komplex, da sich beide Bereiche nicht klar voneinander trennen lassen und so eine Validierung erschwert wird. Das Bewusstsein für dieses Problem wird jedoch oft in den Schlussfolgerungen in der Literatur deutlich.

Zur Vermeidung dieses Problems existieren grundsätzlich zwei Möglichkeiten. Einerseits die Veränderung der Lehre bzw. des Lehrmaterials zwischen der Durchführung einzelner Erhebungen. Dies erscheint nicht sehr sinnvoll, da hier auch gute und in der Praxis bewährte Materialien geändert werden würden, um ihre Qualität wiederum zu belegen. Andererseits besteht die Möglichkeit des Einsatzes eines Messinstruments in verschiedenen Kursen (z. B. an verschiedenen Universitäten). Da die Forschung zu Fehlvorstellungen in der Informatik jedoch jung ist und bislang nur wenige Messinstrumente in einer Entwicklungsstufe vorliegen, die einen nicht an bestimmte Institutionen gebundenen Einsatz ermöglichen, ist dies schwierig. Andere Fächer zeigen jedoch, dass dies langfristig möglich ist. Es ist daher anzunehmen, dass dies auch in der Informatik gegeben ist.

Ein weiterer Aspekt, der meist nicht berücksichtigt wird, sind die Vorkenntnisse der Probanden. Die in den Studien betrachteten Teilnehmer von Anfangsvorlesungen sind, wie zuvor bereits mehrfach dargestellt, sehr heterogen aufgestellt. Grund dafür sind die unterschiedlichen Vorkenntnisse, die Studierende zu Beginn ihrer Ausbildung durch ihre informatische Vorbildung (durch Schulunterricht und privates

Interesse) haben. Es ist schwierig, Vorkenntnisse von Lernenden in unterschiedlichen Fächern zu vergleichen, aber durch die zum einen uneinheitliche Ausbildung im Schulfach Informatik und zum anderen auch die Attraktivität von informatischen Themen als Freizeitbeschäftigung, ist anzunehmen, dass die Heterogenität in der Informatik im Vergleich deutlich stärker ist (vgl. dazu auch Abschnitt 3.5). Demzufolge muss diesem Thema in der Forschung zu Fehlvorstellungen auch eine besondere Beachtung geschenkt werden.

3.5. Ursachen für die Herausbildung von Fehlvorstellungen

Die Mehrheit der Veröffentlichungen zu Fehlvorstellungen in der Informatik konzentriert sich auf die Erhebung und Messung von beobachteten Fehlvorstellungen. Dies schließt zumeist Hypothesen ein, die sich auf die Ursachen für die jeweiligen Fehlvorstellungen beziehen. Diese werden jedoch nur selten überprüft. Nichtsdestotrotz soll im Folgenden ein Überblick darüber gegeben werden, indem auch nichtinformatische Ergebnisse herangezogen und übertragen werden.

Als Ursache für Fehlvorstellungen lassen sich zunächst einmal fächerübergreifende Faktoren nennen, die zweifellos auch in der Informatik existent sind. Zu diesen gehört vor allem eine unterschiedliche Denkweise bei den Lernenden. Dies bezieht sich sowohl auf den Unterschied zwischen Kindern und Erwachsenen, als auch auf Anfänger und Experten (vgl. die allgemeindidaktische Definition von Fehlvorstellungen in Kapitel 2) und die bei diesen Gruppen vorhandenen Vorerfahrungen. Des Weiteren lassen sich in den hier genannten Publikationen und Studien auch Besonderheiten für das Fach Informatik identifizieren, die im Folgenden gesondert beschrieben werden sollen.

3.5.1. Vorerfahrungen und Verknüpfung von Vorwissen

Eine häufig genannte Ursache für die Herausbildung von Fehlvorstellungen sind Vorerfahrungen mit dem Themengebiet ausserhalb der Schule bzw. der Universität und die Verknüpfung dieses Wissens (d. h. des Vorwissens) mit neu erlernten Inhalten. Diese Erfahrungen erfolgen in der Informatik häufig schon zu einem sehr frühen Zeitpunkt und meist in spielerischer Form. In der 2013 veröffentlichten ICILS Studie lag der Anteil der Achtklässler, die täglich einen Computer benutzen,

bei mehr als 90 %. Das hierfür nötige Wissen wird selten strukturiert erworben, sondern durch eigene Erfahrungen und Beobachtungen („Ausprobieren“) oder andere Nutzer (insbesondere Eltern und Freunde). Wichtig ist, dass Kinder ihr Wissen auf diesem Weg zwar erweitern und ihre Kenntnisse vertiefen, dies aber nicht dazu führt, dass sie ein aus wissenschaftlicher Perspektive besseres Verständnis dafür haben. Dies schließt zwar ein, dass sie Fachbegriffe adaptieren und im richtigen Kontext verwenden, aber ein falsches bzw. unvollständiges Verständnis davon haben (Hammond u. Rogers, 2006, S. 13).

Dieses Phänomen zeigt sich besonders in Studien, die sich mit Computernutzern in dieser Altersklasse beschäftigen. So befragten Hammond u. Rogers (2006) Kinder im Alter von etwa 10 Jahren zu ihren Vorstellungen über Computer und deren Funktionsweise. Hier bestätigt sich, dass häufig aus einfachen Beobachtungen Rückschlüsse auf grundlegende Funktionsweisen gezogen werden. So beschrieben die Teilnehmer, dass sich eine Maus auf der rechten Seite des Computers befinden muss, da sie aufgrund ihrer Beobachtung auf die Allgemeinheit schließen. Auch wussten viele Kinder zwar, dass der Computer über das Modem mit dem Internet verbunden ist, gingen aber gleichzeitig davon aus, dass Webseiten im Modem gespeichert sind. Diese intuitiven Modelle haben starke Ähnlichkeit mit aus anderen Disziplinen bekannten Modellen wie z. B. dem häufig beobachteten Phänomen, dass Kinder zwar wissen, dass die Erde rund ist, andere Fragen, die sich nur indirekt auf die Form beziehen, aber so beantworten, als ob die Erde flach wäre (Nussbaum u. Novak, 1976, zitiert nach Vosniadou u. Brewer, 1992).

Nicht nur die Vorerfahrung als Computernutzer, sondern auch die intensivere Beschäftigung mit der Thematik in der Freizeit ist ein weiterer Faktor bei der Entwicklung von Vorwissen und damit auch eine potentielle Ursache für Fehlvorstellungen. Programmieranfänger, die erste Erfahrungen ausserhalb der Schule, des Studiums oder ähnlichen konzeptbasierten Kursen machen, tun dies häufig unstrukturiert, indem sie benötigte Informationen selektiv in Lehrbüchern oder dem Internet suchen. Konzepte werden dann erlernt, wenn sie zur Lösung eines Problems benötigt werden, ohne dass tatsächlich nachvollzogen wird, ob ein bestimmtes Vorwissen nötig ist. Dies kann zum einen dazu führen, dass die genutzten Informationen auf einzelne Codezeilen reduziert sind (im Regelfall durch das Kopieren von Beispielen per Copy & Paste) und die tatsächliche Funktion intuitiv darauf aufbauend erschlossen wird. Zum anderen entstehen auf diesem Weg Wissenslücken, die ebenso durch Beobachtung geschlossen werden. Dies kann zu falschen oder unvollständigen mentalen Modellen und somit zu Fehlvorstellungen führen (vgl. Kapitel 2).

Im Allgemeinen ist jedoch festzustellen, dass Fehlvorstellungen in der Informatik deutlich weniger durch Vorerfahrungen und Vorwissen gefestigt sind als in anderen Fächern. Während Kinder schon früh mit physikalischen Gesetzmässigkeit in Berührung kommen und Präkonzepte entwickeln (vgl. Unterabschnitt 2.3.3), entstehen z. B. Fehlvorstellungen über Pointer oder Rekursion erst bei der direkten „Konfrontation“ mit der Informatik. Dies findet zwangsläufig erst zu einem späteren Zeitpunkt des strukturierten Lernprozesses statt.

3.5.2. Falsche oder unvollständige Analogien und Modelle in der Lehre

Zwar ist es nur schwerlich möglich, einzelne Fachgebiete hinsichtlich der in der Lehre verwendeten Analogien und Modelle zu vergleichen, aber die Informatik ist zweifellos ein Fach, in dem diese eine sehr große Rolle spielen. Insbesondere in den Anfangskursen werden viele Themen und Konzepte mit bekannten Metaphern oder Analogien eingeführt. Ein häufiges Beispiel hierfür sind Matroschkas (vgl. Abschnitt 3.3.2), die zur Erklärung der Rekursion genutzt werden oder das Behältermodell zur Funktionsweise von Variablen. Studien haben gezeigt, dass sich gute und vorsichtig ausgewählte Analogien positiv auf viele Faktoren der Lehre und des Lernprozesses in der Informatik auswirken, wie z. B. das Verständnis von Programmcode und die Bearbeitungszeit von Programmieraufgaben (vgl. Forišek u. Steinová, 2012).

Wie und nach welchen Kriterien eine Analogie tatsächlich „gut“ ist bzw. als „gut“ bewertet werden kann, lässt sich nur schwer definieren. Die Hauptgefahr besteht darin, dass zwar die Analogie als solches nachvollzogen wird, das dahinterstehende Konzept aber nicht angewandt werden kann. Gleichzeitig können auch Schlussfolgerungen zu weiteren Aspekten gezogen werden, die nicht Inhalt der tatsächlichen Erklärung waren. Dies kann dazu führen, dass sich die einmalig falsch adaptierte Analogie weiter durch den Lernprozess fortsetzt.

Auch viele Lehrmittel basieren darauf, dass sie nicht nur die Thematik in theoretischer Form darstellen, sondern diese visualisieren oder in analoger Form präsentieren (wie z. B. Computer Science Unplugged). Dies unterscheidet die Informatik insofern von vielen naturwissenschaftlichen Disziplinen, dass die durchgeführten Experimente hinsichtlich ihrer Kernaussage nicht konkret sind. So können insbesondere in der Chemie und Physik Experimente tatsächlich mit den jeweiligen „Materialien“ durchgeführt werden, während dies in der Informatik nahezu unmöglich ist, da

Konzepte nicht direkt beobachtet werden können. So kann zwar z. B. ein Suchalgorithmus Zeile für Zeile nachvollzogen werden, aber der tatsächliche Ablauf kann nur dargestellt werden, wenn er visualisiert wird. Dies bedeutet nicht zwangsläufig einen Nachteil, unterscheidet die Informatik aber in diesem Punkt von vielen Konzepten in anderen naturwissenschaftlichen Fächern.

Analogien in der Informatik müssen nicht zwangsläufig bewusst gelehrt werden, sondern ergeben sich auch durch die natürliche Sprache. Dies ist in besonderem Maße daran geknüpft, dass die Informatik zahlreiche Begriffe aus der Alltagssprache und anderen Fachdisziplinen übernommen hat, die konzeptionell ähnlich sind. Eine Metapher wie eine *Maus* sollte dabei nicht zu Verwechslungen führen. Bei Konzepten wie *Schleifen* oder *Klassen* kann eine falsche Schlussfolgerung aber zweifellos zu Fehlvorstellungen führen.

3.6. Fehlvorstellungen und ihre Messung in der Praxis

In den letzten Jahren hat der Wunsch nach einem Messinstrument für die Informatik, bzw. auch Messinstrumenten für verschiedene Disziplinen der Informatik, stark zugenommen. Das gewünschte Ziel dieser Instrumente ist es, das Wissen von Anfängern zuverlässig zu messen und auch Fehlvorstellungen dabei identifizieren zu können. Dies geht besonders darauf zurück, dass diese Instrumente in vielen MINT-Fächern („STEM disciplines“, z. B. Tew u. Guzdial (2010)) existieren und erfolgreich eingesetzt werden. Die Zusammenstellung eines solchen hat sich allerdings als sehr schwierig erwiesen. Im Folgenden soll ein kurzer Überblick über die Eigenschaften existierender Messinstrumente gegeben werden.

3.6.1. Repräsentativität der Inhaltsbereiche

Ein Aspekt, der sich hauptsächlich auf Fehlvorstellungen in der Programmierung beschränkt, ist ihre Darstellung und Spezifikation. Eine Analyse bereits veröffentlichter Testinstrumente zur Identifikation von Fehlvorstellungen hat gezeigt, dass diese mehrheitlich Aufgaben auf Basis einer bestimmten Programmiersprache enthalten. Kenntnisse dieser sind somit zur Bearbeitung der Aufgaben dringend nötig. Dies führt zu der Frage, inwieweit die identifizierten und gemessenen Fehlvorstellungen signifikant im Sinne einer grundlegenden Fehlvorstellung in der Informatik sind.

Dies ist bei einigen bekannten Fehlvorstellungen offensichtlich (z. B. das Verständnis von Datentypen in Python), bei anderen jedoch nur durch Testergebnissen und Studien zu erschließen. So bewerten Lehrende die Schwierigkeit einzelner Konzepte je nach Ausrichtung ihres Kurses unterschiedlich (Schulte u. Bennedsen, 2006). Dementsprechend muss auch hinterfragt werden, ob Fehlvorstellungen, die zunächst einmal unabhängig von einer Programmiersprache sind, überhaupt bei Lernenden unterschiedlicher Programmiersprachen entstehen. Treten bestimmte Fehlvorstellungen z. B. nur bei Lernenden in einem Java-Kurs auf, bei Lernenden in einem C++-Kurs jedoch nicht?

Die Erstellung eines einheitlichen, kurs- und sprachenübergreifenden Messinstrumentes zur Überprüfung möglicher Fehlvorstellungen von Lernenden wird zwar häufig als Ziel des Entwicklungsprozesses genannt (z. B. Goldman u. a., 2008), ist aber aus Sicht der Autorin dieser Arbeit nicht realistisch. Gleichermaßen ist dies auch nicht wünschenswert, da ein solches Vorgehen zwangsläufig dazu führt, dass häufige, aber auf eine individuelle Programmiersprache, Entwicklungsumgebung oder ähnliche Themen bezogene Fehlvorstellungen ausgeschlossen werden müssen, da diese das Instrument ansonsten zu umfangreich werden lassen würden.

Der Umstand, dass Inhalte informatischer Einführungskurse sehr unterschiedlich sind, führt auch zur zuvor genannten Problematik der kleinen Messgruppen: um initiative Forschungsansätze auf eine größere Gruppe anzuwenden, bedarf es häufig umfangreicher Anpassungen, die wiederum die Vergleichbarkeit der Ergebnisse gefährden. So können durch eine „Übersetzung“ von Programmcode in eine andere Programmiersprache unterschiedliche Fehlvorstellungen in der Messung auftreten. Generell existieren weitere Einflussfaktoren, die zu einem vermehrten Auftreten von Fehlvorstellungen führen können (z. B. die Verwendung ungeeigneter Analogien), wenn Studien sich auf einzelne Kurse oder eine geringe Anzahl von Kursen beziehen.

Als ein Grund für die Vielfalt der Themen und Konzepte in Einführungskursen in der Informatik wird die schnelle Entwicklung des Faches genannt. Während sich die grundlegenden Konzepte in vielen Teildisziplinen der Physik in den letzten 50 Jahren nur wenig gewandelt haben (z. B. in der Mechanik oder der Elektrodynamik), haben sich Inhalte der Informatik stark verändert - grundlegende Paradigmen und Konzepte wie die Objektorientierung sind heute essentiell, während algebraische Grundlagen heute nur noch am Rande vermittelt werden. Diese Entwicklung scheint sich allerdings in den letzten Jahren zu verlangsamen. So wird deutlich, dass immer mehr Modelle entworfen werden, die Regeln für die Erstellung von Curricula definieren, anstatt lediglich neue Themen zu generieren (vgl. z. B. Hazzan u. a.,

2008). Es bleibt abzuwarten, ob sich auch in der Informatik ein fester Stamm an Konzepten langfristig etabliert und somit die Messbarkeit des Wissens zu diesen Konzepten verbessert wird.

3.6.2. Instrumente zur Messung von Fehlvorstellungen

Viele publizierte Studien, die sich mit der Messung von Fehlvorstellungen in der Informatik beschäftigt haben, enthalten eine implizite oder explizite Aufforderung an Lehrende, diese aktiv in der Lehre zu nutzen. Damit soll die Existenz von Fehlvorstellungen durch zielgerichtete Aufgaben geprüft und gegebenenfalls im Anschluss korrigiert werden. Alternativ sollen häufige Fehlvorstellungen vorgestellt werden. Bemerkenswert ist, dass hierzu nur in einzelnen Fällen Material bereitgestellt wird, bzw. die Forschungsergebnisse nicht vollständig veröffentlicht werden. Begründet wird dies gerne mit der Gefahr, dass Instrumente so auch für Lernende verfügbar sind und diese sich auf einen Test vorbereiten können. Dies ist aus Sicht der Autorin der vorliegenden Arbeit jedoch zu vernachlässigen. Zum einen befindet sich die Mehrheit der Instrumente in einem Entwurfsstatus. Selbst wenn sie bereits in mehreren Schritten erprobt und überarbeitet wurden, so wird häufig darauf hingewiesen, dass sie einer weiteren Überarbeitung oder Anpassung bedürfen, um auch breiter eingesetzt zu werden. Zum anderen stellt sich die Frage, ob auf Seiten der Testteilnehmer überhaupt ein Anreiz besteht, im Test die richtige Antwort zu geben, da diese meist unabhängig von der Notengebung im jeweiligen Kurs sind.

3.7. Fehlvorstellungen als Thema in der Ausbildung von Informatiklehrkräften

Um einen Überblick darüber zu erhalten, wie in der Praxis mit Fehlvorstellungen umgegangen werden sollte, muss auch die Ausbildung von Lehrkräften betrachtet werden. Zwar ging man lange davon aus, dass Lernende vor Beginn der strukturierten Ausbildung keinerlei Kenntnisse und Vorwissen haben und legte die Planung des Unterrichts darauf aus, das Wissen aufzubauen, doch hat in den letzten Jahrzehnten ein starkes Umdenken in den naturwissenschaftlichen Fächern stattgefunden (Barke, 2006). So existiert in der Biologie, der Chemie und auch der Physik mittlerweile eine breite Sammlung an Lehrmaterialien zu Fehlvorstellungen, die entweder als Kapitel ein fachdidaktischen Werken oder sogar als eigenständiges Buch veröffentlicht werden.

Leider findet das Thema Fehlvorstellungen dagegen nur geringe Beachtung in der universitären Ausbildung von Informatiklehrkräften in Deutschland. Dies bestätigt sich sowohl in den Veranstaltungsunterlagen, als auch in den informatikdidaktischen Lehrbüchern. Diese erwähnen zwar die Möglichkeit von Fehlvorstellungen, es fehlt hier jedoch eine einheitliche Strukturierung, wie sie in anderen Fächern zu finden ist. Dies bezieht sich insbesondere auf die Präsentation und Bereitstellung von konkreten praxisrelevanten Beispielen.

Smith et al. nennen die Publikation von Fehlvorstellungen als Hauptaufgabe der Forschung in diesem Gebiet:

> „A major task for research in mathematics and science learning is to document misconceptions in as many subject-matter domains as possible.“ (Smith u. a., 1993, S. 16)

Zweifellos ist allerdings auch die fehlende Forschung bzw. die fehlende Synopse der Forschungsergebnisse zu Fehlvorstellungen ein Grund dafür, warum diese nur selektiv betrachtet werden. Ein Indiz hierfür ergibt sich besonders aus dem Vergleich von Lehrmaterialien der Informatik und anderen mathematischen und naturwissenschaftlichen Disziplinen. Der Autorin dieser Arbeit wurde im Verlauf des Forschungsprozesses mehrfach auf nationalen und internationalen Konferenzen der Wunsch herangetragen, die hier gesammelten Fehlvorstellungen in der Informatik zur Verfügung zu stellen. Hier scheint somit ein großer Bedarf zu existieren, der erfüllt werden sollte (vgl. Kapitel 7).

Ebenso existieren in der Informatik im Gegensatz zu vielen anderen Fächern nur wenige grundlegende Veröffentlichungen, die einen Überblick über bereits dokumentierte Fehlvorstellungen geben. Abgesehen von der objektorientierten Programmierung wurden Forschungsergebnisse nur selten erneut aufgegriffen und weiterentwickelt. Sorva (2012) ist es im Rahmen seiner Dissertation allerdings sehr erfolgreich gelungen, dies für die objektorientierte Programmierung zu tun. Er hat dazu insgesamt 162 Fehlvorstellungen aus der Literatur und aus der eigenen Forschung in elf Kategorien eingeordnet. Diese wurden in eine visuelle Programmierumgebung integriert, um Fehlvorstellungen auf diesem Weg zu identifizieren.

3.8. Zusammenfassung und Fragestellung im Kontext der Arbeit

Abschließend lässt sich zusammenfassen, dass sich die Forschung zu Fehlvorstellungen in der Informatik in den vergangenen 40 Jahren deutlich entwickelt und sowohl an Qualität als auch an thematischer Breite gewonnen hat. Während zu Beginn der Ansatz wenig didaktisch geprägt war und das Fachwissen nur in einer fortgeschrittenen Phase des Lernprozesse betrachtet wurde, sind es mittlerweile meist Anfänger in Einführungskursen, die Gegenstand der Forschung sind. Nichtsdestotrotz ist es unbestreitbar, dass selbst dieses eher schmale Feld noch viele Lücken hat.

Diese Lücken sind besonders darauf zurückzuführen, dass einzelnen Themen sehr große Aufmerksamkeit zukam, während andere nahezu vollständig ignoriert wurden. Dies geht einher mit einer sowohl auf fachwissenschaftlicher, als auch auf fachdidaktischer Ebene vorhandenen großen Heterogenität.

Ein weiterer Aspekt, der insbesondere beim Vergleich mit anderen Fächern deutlich wird, ist die fehlende konsequente Verfolgung von Forschungszielen und die selten erfolgte Adaption von vorhandenen Forschungsergebnissen. Während der Recherche für die in diesem Kapitel erfolgte Analyse stieß die Autorin auf mehrere Publikationen, in denen initiative Ansätze beschrieben und das weitere geplante Vorgehen vorgestellt wurde. Die Suche nach entsprechenden Folgeergebnissen verlief allerdings ins Leere. In einzelnen Fällen wurde die Beendigung der Forschungsvorhaben aufgrund verschiedener Herausforderungen und nicht zu lösender Probleme durch die Autoren bestätigt.

Insbesondere die in den letzten Jahren begonnene Entwicklung von CIs in Teildisziplinen der Informatik ist allerdings ein Beleg dafür, dass wertvolle Ergebnisse vorhanden sind, die allerdings weiter strukturiert und geordnet werden müssen, um den Status eines allgemein nutzbaren Instruments zu erlangen. Gleichzeitig fehlt leider oft in der Praxis die Bereitschaft, diese einzusetzen, aber auch das Wissen darüber, dass diese Messinstrumente überhaupt existieren. Letzteres ist wiederum darauf zurückzuführen, dass bereits entwickelte CIs nicht vollständig publiziert wurden und damit nicht frei verfügbar sind.

Allgemein mangelt es in der Informatik am grundlegenden Bewusstsein für Fehlvorstellungen, das sich insbesondere in der Lehrerausbildung widerspiegelt. Fehlvorstellungen werden dort kaum oder selten thematisiert. Aus Erfahrung der Autorin dieser Arbeit besteht allerdings ein großes Interesse der Lehramtsstudierenden an diesem Thema. In der Diskussion mit Studierenden kam schnell die Frage auf,

inwiefern sich die Informatik von der Mathematik unterscheidet, da in letzterer Fehlvorstellungen ein elementarer Bestandteil der Ausbildung sind. Der vermutlich wichtigste Faktor ist hier die junge Geschichte der Lehre in der Informatik und auch der Informatikdidaktik.

Dies ist nicht der einzige Aspekt, der dem jungen Alter der Disziplin zuzuschreiben ist. Die Ergebnisse zahlreicher Forschungsprojekte zeigen, dass zur Ermittlung und Verifizierung von Fehlvorstellungen ein mehrstufiger Prozess durchlaufen werden muss, der sich über mehrere Jahre oder gar Jahrzehnte erstreckt. Dass dies für die Informatik noch nicht umfassend und abschließend geschehen konnte, ist dementsprechend nachvollziehbar.

Die Motivation, Themengebiete umfassend zu beschreiben, scheint in der Forschung deutlich weniger vorhanden zu sein als die Motivation, neue Gebiete zu betreten. Letzteres ist nicht grundsätzlich als unwichtiger oder schlechter zu bewerten. Allerdings muss der Schritt zur Zusammenführung, Vereinheitlichung und Verifizierung verschiedener Forschungsergebnisse zwangsläufig erfolgen, um Fehlvorstellungen z. B. auch in die Ausbildung von Informatiklehrern integrieren zu können und vorhandene Materialien zu entwickeln, die für die Fort- und Weiterbildung genutzt werden können.

Diese Arbeit kann insofern dazu beitragen, als sie, wie in diesem Kapitel geschehen, die Forschungsergebnisse strukturiert, zusammenfasst und daraus resultierende Fragestellungen festhält. Dies kann als eine Art Inhaltsverzeichnis bzw. als Leitfaden für eine Orientierung im Themengebiet dienen. Gleichzeitig konnten im Rahmen der Analyse des Forschungsstandes auch verschiedene Probleme und Besonderheiten festgehalten werden.

Ein wichtiges Forschungsziel dieser Arbeit ist es, Konzepte der Informatik zu identifizieren, die sowohl grundlegend sind als auch häufig mit Fehlvorstellungen von Lernenden verknüpft sind. Dies entspricht dem zuvor vorgestellten Verfahren zur Erstellung von CIs nach Almstrum u. a. (2006), bei dem die zentralen Inhalte des Messinstruments, zu denen im nächsten Schritt Fragen generiert werden, durch vorhandene Expertise generiert werden. Dies sind in diesem Fall vorhandene und publizierte Forschungsergebnisse zu Fehlvorstellungen. Auf diesem Weg ergibt sich eine kleine, aber zielgerichtet strukturierte Sammlung. Diese kann, in leicht bearbeiteter bzw. abgewandelter Form auch in der Ausbildung von Lehrenden eingesetzt werden (vgl. Kapitel 7).

4. Identifikation und Bewertung informatischer Konzepte und der Häufigkeit damit verbundener Fehlvorstellungen

4.1. Einleitung

Nachdem im vorhergehenden Kapitel das Forschungsfeld der Fehlvorstellungen in der Informatik analysiert und bewertet wurde, soll nun ein Vorgehen zur Identifikation relevanter Inhalte in der Informatiklehre, bei denen die Wahrscheinlichkeit für das Entstehen einer Fehlvorstellung besonders hoch ist, erstellt werden. Dies erscheint nötig, um zum einen eine Konzentration auf in der Praxis weniger relevante Konzepte zu vermeiden und zum anderen zu verhindern, besonders interessante Konzepte auszulassen.

In den Forschungsergebnissen zu Fehlvorstellungen, die im vorhergehenden Kapitel vorgestellt wurden, zeigt sich eine besondere Schwerpunktlegung im Gebiet der objektorientieren Programmierung. Zahlreiche Studien beschäftigen sich mit Missinterpretationen von Klassen und Objekten und geben Hinweise, wie diese in der Lehre verhindert werden können. Auch Paul u. Vahrenhold (2013) gehen darauf ein, dass ihr Forschungsfeld (Algorithmen und Datenstrukturen) in der Vergangenheit häufig vernachlässigt wurde und stattdessen die Objektorientierung große Aufmerksamkeit bekam. Die Gründe dafür, warum genau dieses Themengebiet gewählt wurde, werden dabei selten genannt.

Das Interesse an Fehlvorstellungen zur Objektorientierung steht zweifellos in engem Zusammenhang zu der allgemeinen Popularität des Themas, die sich in den

letzten Jahren in der Informatikdidaktik gezeigt hat. Auch wenn die Dominanz darauf hindeuten könnte, lässt dies aber keine Schlussfolgerung zu, dass tatsächlich ein Zusammenhang zur Häufigkeit oder Relevanz der in der Praxis auftretenden Fehlvorstellungen besteht. Aus diesem Grund scheint auch die Konzeption eines Testinstruments zur Messung des Wissens über Fehlvorstellungen auf Basis der vorhandenen Forschungsschwerpunkte nicht angemessen. Hier stellt sich somit zunächst einmal die Frage, nach welchen Kriterien stattdessen vorgegangen werden kann, um letztendlich ein zuverlässiges Testinstrument zu erstellen.

Die Schwierigkeit eines Konzeptes und die Wahrscheinlichkeit für die Entwicklung von Fehlvorstellungen zu diesem Konzept sind in jedem Fall nicht gleichzusetzen. Allerdings ist unbestreitbar, dass hier ein Zusammenhang besteht (vgl. Kapitel 2). So entstehen Fehlvorstellungen auch dadurch, dass herausfordernde Themen durch Lernende nicht vollständig erschlossen werden können und eine Abkürzung gewählt wird, indem voreilige Schlüsse gezogen werden, aus denen eine Fehlvorstellung resultieren kann.

Mit den zuvor beschriebenen Fragestellungen geht auch die Frage einher, welche Themen und Konzepte überhaupt Teil der Informatikausbildung sein sollten und wie diese Inhalte möglichst allgemeingültig zusammengestellt werden können. Nur so kann gewährleistet werden, dass das Instrument nicht nur von einer eingeschränkten kleinen Zielgruppe bearbeitet werden kann. Dazu sollen im folgenden Abschnitt Forschungsansätze und Ergebnisse präsentiert und auf ihre Verbindung zu Fehlvorstellungen untersucht werden.

4.2. Analyse des Forschungsstandes zu fundamentalen Konzepten, Prozessen und Methoden für die Informatikausbildung

Die Frage nach geeigneten Inhalten für die Informatikausbildung und ihrer Auswahl, sowie eine darauf bezogene Diskussion, ist seit Jahren in der informatikdidaktischen Forschung allgegenwärtig. Neben theoriegeleiteten Ansätzen, die eine individuelle, meist subjektive, Bewertung erfordern, gibt es empirische Ergebnisse, die unter Anwendung sehr verschiedener Methoden entstanden sind. Im folgenden Abschnitt sollen mehrere Beispiele hierzu präsentiert und mit besonderem Schwerpunkt auf die Betrachtung von Fehlvorstellungen analysiert werden.

4.2.1. Konzeptbasierte Ansätze

Fundamentale Ideen

Ein vielbeachteter Ansatz zur Identifikation und Kategorisierung sind die fundamentalen Ideen nach Schwill. Aufbauend auf einem didaktischen Prinzip von Bruner, das zur Identifikation grundlegender Strukturen einer Wissenschaft dient, definiert er vier Kriterien für die Informatik:

> Eine **fundamentale Idee** (bezgl. einer Wissenschaft) ist ein Denk-, Handlungs-, Beschreibungs- oder Erklärungsschema, das
>
> 1. in verschiedenen Bereichen (der Wissenschaft) vielfältig anwendbar oder erkennbar ist (**Horizontalkriterium**),
> 2. auf jedem intellektuellen Niveau aufgezeigt und vermittelt werden kann (**Vertikalkriterium**),
> 3. in der historischen Entwicklung (der Wissenschaft) deutlich wahrnehmbar ist und längerfristig relevant bleibt (**Zeitkriterium**),
> 4. einen Bezug zu Sprache und Denken des Alltags und der Lebenswelt besitzt (**Sinnkriterium**). (Schwill, 1993, S. 8)

Schwill definiert zudem drei Masterideen, die er als fundamentale Säulen der Informatik bezeichnet: Algorithmisierung, strukturierte Zerlegung und Sprache (Schwill, 1993, S. 11). Später ergänzt er das Zielkriterium als fünftes Kriterium, das eine Annäherung an eine idealisierte Zielvorstellung beschreibt, die aber möglicherweise unerreichbar ist (Schubert u. Schwill, 2011).

Auch wenn die fundamentalen Ideen der Informatik durch ihre kriteriengeleitete Konzeption eine eindeutige Identifikation nahelegen, so ist diese häufig nur subjektiv möglich. Dies hängt in besonderem Maße in der Auslegbarkeit der Definitionen zusammen. So besteht beispielsweise ein großer Interpretationsspielraum bei der Einschätzung des Vertikalkriteriums, das eine Vermittelbarkeit auf „jedem intellektuellen Niveau“ verlangt. Schubert u. Schwill (2011) erklären dazu beispielhaft, dass schon Kinder durch ihre Eltern die fundamentale Idee der Zugriffskontrolle lernen. Dies trifft im Hinblick auf ein sehr simplifiziertes und didaktisch reduziertes Konzept zweifellos zu: ein Kind versteht, dass ein Schloss nur geöffnet werden kann, wenn der dazugehörige Schlüssel vorhanden ist. Insofern kann dieses Kriterium sicherlich als erfüllt verstanden werden. Nichtsdestotrotz muss hinterfragt werden, ob ein Verständnis auf einem so niedrigen Niveau dem Konzept einer logischen Zugriffskontrolle in der Informatik genügt.

Ähnliches trifft auch auf das Zeitkriterium zu. Schubert u. Schwill (2011) definieren, dass eine Idee, die „in der historischen Entwicklung deutlich wahrnehmbar ist und längerfristig relevant bleibt“ dieses Kriterium erfüllt. Aufgrund der kurzen Geschichte der Informatik und des schnellen Wandels und der Entwicklung von Konzepten und Methoden ist fraglich, ob dies überhaupt zuverlässig möglich ist. Für entsprechende Fehleinschätzungen hinsichtlich der Relevanz, durch die Ideen zunächst als wertvoll eingeschätzt wurden, aber schon wenig später irrelevant waren, existieren zahlreiche Beispiele in der Informatik (vgl. z. B. die Zusammenstellung von Maurer (2000)). Schwill weist dazu jedoch explizit auf die Unterscheidung von Ideen und Details hin: „Während fundamentale Ideen eine längerfristige Gültigkeit besitzen (vgl. Zeitkriterium), sind Details schnell wieder veraltet.“. Auch wenn dies als solches plausibel erscheint, schließt sich hier eine weitere Frage an: woran kann festgemacht werden, ob eine Idee tatsächlich eine Idee ist oder ein Detail? Vermutlich kann eine eindeutige Antwort darauf meist nicht gegeben werden, ohne das jeweilige Konzept konsequent auf eine möglichst allgemeingültige Idee zu reduzieren.

Humbert (2006) sieht insbesondere ein Problem darin, dass der Großteil der von Schwill vorgestellten Fundamentalen Ideen nicht weiter diskutiert wurde, sondern die Erfüllung der Kriterien als gegeben präsentiert wurde. Ihm fehlen Argumente, die dies belegen und dadurch auch das generelle Konzept weiter beleuchten. Zudem würden Beispiele zu Ideen fehlen, für die dies nicht zutrifft. Auch Modrow kritisiert, dass Schwills Ideenkatalog zu umfangreich ist, um tatsächlich den Kern des Faches zu definieren, hält das Konzept der fundamentalen Ideen aber gleichzeitig als Klassifikationsschema für geeignet (Modrow, 2003).

Zusammenfassend zeigen die zuvor genannten Punkte, dass es sich bei den fundamentalen Ideen um ein sinnvolles Konzept handelt, um die zentralen Ideen eines Faches zu identifizieren und auch eine Leitlinie für zukünftige Einschätzungen zu geben. Gleichzeitig darf allerdings sowohl aufgrund der Subjektivität einiger Kriterien, als auch aufgrund der nicht vorhandenen Beständigkeit informatischer Themen, nicht außer Acht gelassen werden, dass sie lediglich einen groben Filter darstellen. Im Kontext dieser Arbeit wird deutlich, dass die fundamentalen Ideen lediglich die Relevanz abbilden und keinerlei Information darüber geben, an welcher Stelle sie im Lernprozess stehen oder ob sie allgemein als herausfordernd angesehen werden.

Great Principles of Computing

Auch Denning (2003, 2004, 2005) beschäftigte sich mit der Frage, welche Prinzipien die Säulen der Informatik darstellen. Sein Ziel war es, Konzepte zu identifizieren, die ähnlich wie Photonen und Elektronen in der Physik, das Fachgebiet der Informatik definieren und strukturieren. Er entwickelte zunächst einen prinzipienbasierten Ansatz mit fünf Blickwinkeln auf die Informatik und erweiterten diesen später zu sieben „großen“ Prinzipien: *Computation*, *Communication*, *Coordination*, *Recollection*, *Automation*, *Evaluation* und *Design*.

Im Gegensatz zu Schwill ist Dennings Vorgehen nicht kriteriengeleitet, aber im Ergebnis sind sich beide Ansätze ähnlich. Ein großer Unterschied ist allerdings, dass Dennings Ansatz weniger auf die praktische Lehre ausgerichtet ist und vielmehr eine Sichtweise auf die Informatik definiert. Er versucht damit nicht zu formulieren, was ein Lernender im einzelnen über die Informatik wissen sollte, sondern woraus sie besteht. Es ist damit ein Rahmen, um Themen einzuordnen und kein Instrument, um bedeutsame Themen herauszufiltern.

Threshold Concepts

Der in Abschnitt 3.4.2 bereits beschriebene Ansatz der Threshold Concepts hat gezeigt, dass auf diesem Weg zentrale Konzepte definiert werden können, die zudem als besonders herausfordernd für Lernende angesehen werden. Hier vereinen sich somit die Relevanz und die Herausforderung, die aus Perspektive der Lernenden beurteilt wird. Die Hürde, die hier ebenso wie in anderen Ansätzen besteht, ist allerdings die eindeutige Identifikation. Die bereits vorgestellten von Meyer u. Land (2013) definierten Kriterien sind vage und nicht eindeutig (dies zeigt besonders die mehrfache Verwendung des Begriffs „likely“).

Der Vergleich der Kriterien für Threshold Concepts mit den Kriterien der Fundamentalen Ideen zeigt mehrere grundlegende Unterschiede. So ist eine Fundamentalen Idee per Definition des Sinnkriteriums aus dem Alltag und der Lebenswelt bekannt. Ein Threshold Concept ist hingegen ein Fachkonzept. Des Weiteren kann eine Fundamentalen Idee auf jedem intellektuellen Niveau vermittelt werden. Für viele Threshold Concepts gilt dies explizit nicht (Eckerdal u. a., 2006). Dies macht die Ansätze konträr, allerdings nicht widersprüchlich.

Eine Zuordnung beispielhafter Fundamentalen Idee zu Threshold Concepts verdeutlicht schnell, dass diese häufig mehrfach darin enthalten sind, d. h. ein Threshold

Concept beinhaltet in der Regel mehrere Fundamentale Ideen (Sorva, 2010). Dies kann so interpretiert werden, dass die Kenntnis verschiedener Fundamentaler Ideen zum Verständnis eines Threshold Concepts führt. Dies gibt auch einen Erklärungsansatz, warum letztere häufig eine deutlich größere Hürde darstellen: das erworbene Wissen in Form von Ideen wird gesammelt, zusammengesetzt und auf das Fachkonzept übertragen. Dass in diesem Fall die Wahrscheinlichkeit für die Entstehung von Fehlvorstellungen groß ist, ist offensichtlich.

Rountree u. a. (2013) veröffentlichten das Knowledge, Strategies and Models Framework (KSM Framework), das in Abbildung 4.1 zu sehen ist. Sie stellen damit dar, dass Threshold Concepts die drei Domänen *Knowledge*, *Strategies* und *Models* überspannen, während die fundamentalen Ideen, ebenso wie Fakten und Konzepte, lediglich zum Bereich *Knowledge* gehören. Sie erklären diese deutlich gewichtigere Positionierung der Threshold Concepts durch die Eigenschaft, dass sie aufgrund der tiefgreifenden Veränderung, die sie im Lernprozess widerspiegeln, auch grundlegende Strategien und mentale Modelle ändern oder sogar eine Neuentwicklung anstossen können. Aus Sicht der Autorin dieser Arbeit zeigt sich in dieser Einordnung eine sehr enge Interpretation der Fundamentalen Ideen, die in dieser Form zu streng angesetzt ist. So definiert Schwill selbst die Fundamentalen Ideen als „grundlegende Prinzipien, Denkweisen und Methoden“ (Schwill, 1993, S. 1). Zwar führt er nicht weiter aus, in welcher Ausprägung *Denkweisen* und *Methoden* vorhanden sein müssen, um als Fundamentalen Idee zu gelten. Es ist aber anzunehmen, dass es hier starke Überschneidungen mit den *Strategies* und *Models* nach Rountree u. a. (2013) gibt. So ist die fundamentale Idee der Rekursion zweifellos nicht nur als pures Wissen einzuordnen, sondern stellt auch eine Strategie und ein Modell dar. Nichtsdestotrotz spiegelt diese Einordnung auch die zuvor erwähnte Kritik an Schwills Modell wider: auch wenn ein Konzept eindeutig als fundamentale Idee identifiziert werden kann, gibt dies keine Auskunft darüber, wo dieses tatsächlich in der Informatikausbildung positioniert ist, wie es dort eingesetzt werden sollte und in welchem Kontext es steht.

Zusammenfassung

An dieser Stelle lässt sich zusammenfassen, dass die zuvor genannten konzeptbasierten Ansätze sich zwar in vielen Punkten voneinander abgrenzen bzw. auch bewusst abgegrenzt werden, sich allerdings nicht widersprechen. So sind beispielsweise Ideen, die Schwill als Unterkategorien seiner Masterideen definierte, auch als Threshold Concept bekannt. Ein Vergleich, der die Ansätze vereint und mit Hilfe konkreter

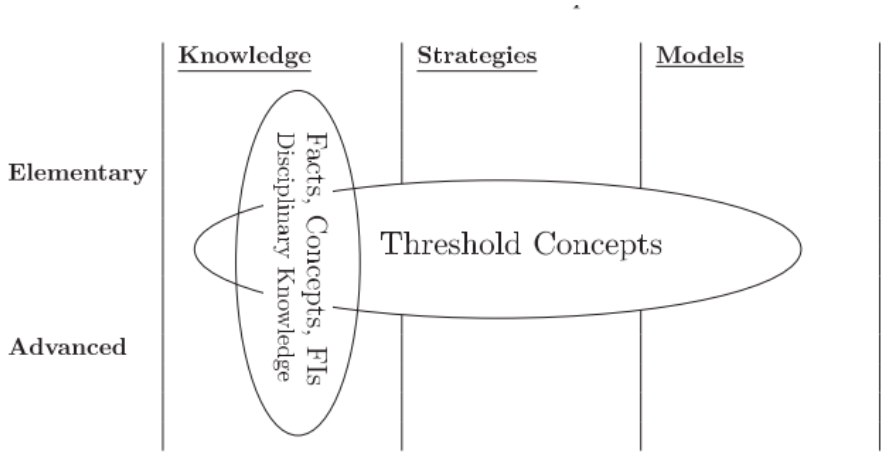


Abbildung 4.1.: KSM Framework nach Rountree u. a. (2013)

Beispiele analysiert, verspricht interessante Ergebnisse, ist aber aufgrund der sehr theoriegeleiteten Konzeption beider Ansätze nur schwer umzusetzen bzw. darf einer Verifizierung in mehreren Einzelschritten.

Grundlegende Prozesse in der Informatiklehre

Neben der Identifikation von relevanten Inhalten der Informatik sind in den letzten Jahren auch Prozesse und Methoden für den Informatikunterricht in den Fokus gerückt. Als ein Beispiel seien hier Prozesse in der Programmierung genannt, deren Relevanz von Bennedsen u. Caspersen wie folgt beschrieben wird:

> „Revealing the programming process to beginning students is important, but traditional static teaching materials such as textbooks, lecture notes, blackboards, slide presentations, etc. are insufficient for that purpose. They are useful for the presentation of a product – a finished program– but not for the presentation of the dynamic process used to create that product.“ (Bennedsen u. Caspersen, 2008)

Sie versuchen hier nicht das Wissen zu definieren, das im Rahmen eines Programmierkurses vermittelt werden sollte, sondern Prozesse, die in einem Einführungskurs (CS1) vermittelt werden sollten. Zu diesen gehört beispielsweise die Nutzung einer IDE und das Testen. Auch wenn dieser Ansatz vielversprechend klingt, ist er leider nur ansatzweise weiterverfolgt worden. Insbesondere fehlt eine theoretische

Fundierung, durch die die durch die Beobachtung ermittelten Prozesse bestätigt und weiter definiert werden konnten. Eine empirische Studie, in der Experten für die Informatik relevante Prozesse bewerten liessen, wurde unabhängig von dieser Beobachtung erstellt. Diese soll auch im nachfolgenden Unterabschnitt 4.2.2 beschrieben werden.

4.2.2. Empirische Ergebnisse

Zendler u. Spannagel publizierten mehrere Erhebungen, in denen sie Konzepte und Prinzipien von Experten der Informatik bewerten haben lassen und somit ihre Relevanz empirisch überprüft und validiert haben (Zendler u. Spannagel, 2008; Zendler u. a., 2008, 2010, 2011).

In einer ersten Studie wurden die zentralen Konzepte aus dem ACM Computing Classification System (in der Ausgabe von 1998) und dem Computer Engineering Curriculum (The Joint Task Force on Computing Curricula u. a., 2004) extrahiert und in einer Umfrage von 37 Professoren deutscher Universitäten anhand der Kriterien der fundamentalen Ideen bewertet (Zendler u. Spannagel, 2008). Dabei wurden die folgenden 15 Konzepte als zentral und besonders relevant identifiziert: *algorithm, computer, data, problem, information, system, language, program, test, communication, software, process, model, structure* und *computation.*

In einer darauffolgenden Erhebung ergänzen Zendler u. a. diese Daten um zentrale Prozesse, um der Forderung nachzukommen, auch diese in die Curricula zu integrieren (Zendler u. a., 2008). Hier wurden, analog zur zuvor genannten Studie, 44 allgemeine Prozesse aus der Bildungsforschung (Costa u. Liebmann, 1996) von 24 Professoren deutscher Universitäten nach den Kriterien der fundamentalen Ideen bewertet. Als Ergebnis wurden die Prozesse *problem solving and problem posing, classifying, finding relationships, investigating, analyzing and generalizing* als besonders bedeutend für die Informatik identifiziert.

Von Zendler u. a. (2011) wurden beide Studien zusammengeführt, indem die identifizierten 16 Konzepte und 15 Prozesse in einer Matrix aufgestellt wurden. Jede Kombination wurde nun von 24 Professoren jeweils auf einer Skala von 1 („nicht relevant“) bis 5 („relevant“) bewertet. Dabei erhielten das Inhaltskonzept *problem* und das Prozesskonzept *analyzing* jeweils (im Sinne der Mittelwerte aller möglichen Konzepte bzw. Prozesse) und auch in Kombination die höchsten Werte. Die Resultate in der Gesamtheit bestätigen außerdem Themenbereiche des K-12 Curriculums wie z. B. *problem solving und algorithmic thinking* (Zendler u. a., 2011, S. 394).

Einen ähnlichen Ansatz verfolgten Goldman u. a. (2008). Sie ermittelten mit Hilfe eines Delphi-Prozesses die Wichtigkeit und Schwierigkeit der Themen in drei Einführungsveranstaltungen in der Informatik: Diskrete Mathematik („discrete mathematics“, Programmiergrundlagen („programming fundamentals“) und Logisches Design („logic design“). Diese Themen dienen als Basis für ein Concept Inventory (vgl. Abschnitt 3.4.1), das eingesetzt werden kann, um grundlegende Kenntnisse zu messen und die Ergebnisse vergleichbar zu machen. Der Delphi-Prozess bestand aus vier Phasen: zunächst wurden Experten der Fachgebiete gebeten, jeweils 10-15 Konzepte zu nennen, die sie für wichtig und schwierig halten. Diese wurden codiert und zu einer Liste zusammengefügt, die im zweiten Schritt von den bereits zuvor beteiligten Experten hinsichtlich drei Kriterien bewertet wurden: Wichtigkeit, Schwierigkeit und erwarteter Beherrschungsgrad („expected mastery“, d. h. das Maß, zu dem ein Student das Thema am Ende des Einführungskurses beherrscht). Da die Bewertungen der Wichtigkeit und des Beherrschungsgrades stark korrelierten, wurden letzterer aus den beiden folgenden Schritten ausgeschlossen. In der dritten Phase wurde der Durchschnitt und der Innerquartilbereich berechnet. Diese Werte wurden den Experten nun vorgelegt und sollten erneut bewertet sollten. Wenn ihre Einschätzung vom durchschnittlichen Ratingergebniss abwich, sollten sie diese begründen. Diese Begründungen wurden anschließend zusammen mit den Durchschnittswerten in der vierten Phase noch einmal an die Experten weitergegeben und von diesen bewertet. Im Bereich der Programmiergrundlagen gab es einen sehr hohen Konsens hinsichtlich der Wichtigkeit von „Difference between Classes and Objects“, das hier auch die höchste Wertung (10 von 10) bekam. Bei „Issues of Scope, local vs. global“ gab es hingegen den größten Konsens hinsichtlich der Schwierigkeit. Den höchsten Wert in diesem Kriterium (9,5) erhielt die Vererbung („inheritance“) (Goldman u. a., 2008, S.258). Im Gebiet des „logic design“ war die Standardabweichung der einzelnen Themen deutlich höher. Hier wurden „State transitions“ und „Verbal Specifications to State Diagram“, (jeweils mit einer Standardabwichung von 0,4) als wichtigste Themen bewertet. Die höchste Schwierigkeit wurde „Debug, Test, Troubleshoot“ zugeordnet (Goldman u. a., 2008, S.260).

Als Gesamtergebnis wurde ein Ranking aus allen Bereichen und den beiden Bewertungskriterien erstellt, in dem die folgenden Themen die ersten zehn Plätze belegten: *procedure design, conceptualize problems and design solutions, abstraction/pattern recognition and use, designing tests, debugging and exception handling, issues of scope (local vs. global), functional decomposition and modularization, inheritance, parameter scope and use in design, memory model* und *references/pointers*. Im Gegensatz zur Studie von Zendler u. a. (2011) wurde neben der Bedeutung der Konzepte auch ermittelt, ob die Konzepte eine besondere Herausforderung für

die Lernenden darstellen. Hier ist interessant, inwieweit sich beide Bewertungen gegenseitig beeinflussen können: ein Konzept, das sich nicht im Rahmen eines Einführungskurses vermitteln lässt, hat auch laut der Studie eine deutlich geringere Bedeutung.

Die Frage der Aussagekraft der Ergebnisse und ihrer Übertragbarkeit auf andere Kurse, Universitäten oder Curricula stellt sich fast automatisch bei der Betrachtung der Ergebnisse. So wurde nur ein Konzept aus dem Bereich der objektorientierten Programmierung in das Ranking der oberen elf Konzepte aus dem Gebiet der Programmiergrundlagen aufgenommen. Goldman u. a. (2008, S.258) betonen gleichzeitig aber, dass ein daraus entstehendes Content Inventory breit eingesetzt werden kann. Dies bedeutet im Umkehrschluss allerdings auch, dass Konzepte der objektorientierten Programmierung, die ein grundlegendes Thema in Einführungsveranstaltungen an Universitäten ist, in diesem Content Inventory nicht behandelt würden. Das Ziel, ein möglichst umfasendes Instrument zu haben, kann somit mit dieser Ausgangslage nicht erreicht werden. So wurde das in Forschungsergebnissen zu Fehlvorstellungen in der objektorientierten Programmierung vielfach genannte Konzept „Difference between Classes and Objects“ auf diesem Wege, obwohl es eine Wichtigkeit von 10.0 erhalten hatte, ausgeschlossen. Eine Ursache hierfür liegt darin, dass der thematische Umfang sehr groß ist und so zwangsläufig nicht alle Themenbereiche erfasst werden können. Gleichzeitig bestätigt dies aber auch ein weiteres Mal, wie problematisch es ist, allgemeingültige Fehlvorstellungen zu identifizieren bzw. diese umfassend in einem Content Inventory abzubilden, ohne ein viel zu umfangreiches Messinstrument zu erstellen. Dies wäre lediglich möglich, wenn zuverlässig Zusammenhänge zwischen Konzepten identifiziert werden können, die belegen würden, dass bei Beherrschung eines bestimmten Konzeptes auch ein anderes verstanden wurde. Dies ist jedoch zum jetzigen Zeitpunkt illusorisch.

Gleichzeitig muss hier auch die Frage nach der Behandlung der Kategorie „Wichtigkeit“, insbesondere hinsichtlich des Verlaufs der gesamten Erhebung, gestellt werden. Die zu Anfang erstellte Liste, auf deren Basis die Bewertung erfolgt ist, basiert auf den subjektiven Meinungen der Experten, die diese im ersten Schritt des Verfahrens genannt haben („[. . .] list 10 to 15 concepts that they considered both important and difficult in a first course in their subject [. . .]“). Auf den gesamten Ablauf der Studie übertragen bedeutet dies, dass ein Kreis von Experten die Wichtigkeit von Konzepten bewertet, die sie selbst zuvor als wichtig nominiert haben. Diese Tatsache relativiert sich zwar dadurch, dass die Experten alle von allen Experten genannten Konzepte bewerten mussten und somit konträre Bewertungen auffallen können (insbesondere wenn ein Konzept nur von einem Experten mit einer hohen Wichtigkeit bewertet wurde). Nichtsdestotrotz entsteht hier eine Art

geschlossener Kreislauf, durch den einige Konzepte eventuell anders eingeschätzt werden, als es außerhalb der Studie geschehen würde. Der hier verfolgte Ansatz ist kein fester Bestandteil des Delphi-Prozesses, sondern ein Vorgehen, das die Forscher selbst gewählt haben. Er lässt sich allerdings dadurch nachvollziehen, da bislang kaum Ergebnisse zur Wichtigkeit von Konzepten existieren (was wiederum auch eine Motivation der Studie ist). Dabei muss bei der Interpretation des Gesamtergebnisses auch das Entstehen von Einschätzungen und Bewertungen beachtet werden. Das Einbringen von Argumenten, um andere von der eigenen Meinung zu überzeugen, ist zwar fester Bestandteil des Delphi-Prozesses, kann aber zu einer Meinungsbildung führen, die von einzelnen Experten gesteuert ist, insbesondere indem sie andere, die keine starke Meinung haben, überzeugen (vgl. Häder u. Häder, 2000). Entsprechende Effekte lassen sich stellenweise in den einzelnen Schritten des Prozesses identifizieren. Goldman u. a. (2008) geben hierzu allerdings keine weiterführenden Informationen.

Shinners-Kennedy u. Fincher (2013) verfolgen das Ziel, Threshold Concepts für die Informatik zu identifizieren. Zu diesem Zweck wurden zunächst durch Umfragen, die im Rahmen von zwei Fachkonferenzen (ITiCSE 2005 und Koli 2005) durchgeführt wurden, 33 Konzepte als mögliche Threshold Concepts nominiert. Den Befragten wurde dabei explizit das Ziel dieser Umfrage und die zu erfüllenden Kriterien für Threshold Concepts dargelegt. Im Anschluss wurden Interviews mit sechzehn Studierenden am Ende ihres Studiums geführt, die u. a. nach Konzepten befragt wurden, die ihnen zunächst Schwierigkeiten bereitet haben. Boustedt u. a. (2007) beschreiben den Ablauf der Interviews im Detail. Zu Beginn wurden die Interviewpartner nach einem Konzept gefragt, bei dem sie selbst Verständnisprobleme hatten. Auf dieses konzentrierten sich die darauffolgenden Fragen, die sich zum einen auf die Kriterien der Threshold Concepts beziehen, zum anderen auf das individuelle Verständnis der Konzepte. Durch die Interviews konnten im Speziellen die objektorientierte Programmierung und Pointer als Threshold Concepts identifiziert werden. Ähnlich wie bei zuvor beschriebenen Ergebnissen erscheint auch hier die Differenzierung der ausgewählten Konzepte diskussionswürdig oder sogar fragwürdig. Die Autoren schlussfolgern selbst, dass eine tiefergehende Analyse der Interviewdaten lohnenswert sein könnte, um spezifischere Threshold Concepts zu identifizieren. Das Vorgehen, sie in einem einzelnen Concept Inventory zu überprüfen, scheint bei einem derart großen Themenkomplex wie der objektorientierten Programmierung eher unrealistisch.

Im Anschluss wurden die Interviews auf Hinweise analysiert, die die Schwelle zum Verstehen des Konzeptes näher definieren können. Um die Nominierung der objektorientierten Programmierung als Threshold Concept zu validieren, wurde

eine Aufgabe entworfen, bei der Programmieranfänger eine Concept Map mit Alltagskonzepten vorgelegt bekamen und diese anschließend analog für das Konzept der objektorientierten Programmierung erstellen mussten. Hier bestätigten sich bekannte Fehlvorstellungen, es gab jedoch keine weiteren Hinweise auf ein Threshold Concept. In einem nächsten Schritt bekamen Studierende die Aufgabe, eine „transformation biography“ zu erstellen, in der sie ein Konzept auswählen und beschreiben sollten, wie dieses ihre Sicht auf die Informatik und ihre Erfahrung verändert hat. Als Ergebnis dieser Arbeitsschritte wird beschrieben, dass die zu Anfang nominierten Konzepte (objektorientierte Programmierung, Abstraktion und Pointer) weiterhin als mögliche Threshold Concepts gesehen werden. Da hier keine eindeutigen Konzepte identifiziert werden konnten, wurden Interviews nach der Critical Incident Technique (CIT) durchgeführt (Flanagan, 1954). Auch sie führten nicht zu vertiefenden Ergebnissen. Dies war besonders darauf zurückzuführen, dass Studierende keine Konzepte benennen konnten oder den Lernprozess nicht weiter einordnen und beschreiben konnten.

Die Liste der einzelnen Forschungsschritte zeigt, dass das Vorgehen investigativ ist. Eine klare Forschungsmethodik existiert zu Beginn nicht und die einzelnen Untersuchungen, die chronologisch beschrieben werden, stellen vielmehr individuelle Forschungsansätze dar, die vorhandene Ergebnisse verfeinern oder validieren sollen. Dies lässt vermuten, dass der Ablauf in dieser Form auch zu Beginn nicht geplant war. Die Ergebnisse geben zwar Impulse für anschliessende Forschungsvorhaben - ein roter Faden lässt sich allerdings nicht erkennen.

Ioannou u. Angeli (2014) untersuchten in ihrer Erhebung Fehlvorstellungen über Prozessoren (CPU) und veröffentlichten in diesem Zusammenhang eine Tabelle mit Themen, die als besonders schwierig angesehen werden (s. Tabelle 4.1). Diese entstand aus einer Analyse von entsprechenden Publikationen und führt sie letztendlich zur Auswahl des Themas CPU um zu verdeutlichen, dass auch Themen ausserhalb von Programmiersprachen als schwierig gelten. Leider geht aus der Beschreibung nicht klar hervor, auf welcher Basis diese Zusammenstellung erfolgte, sie scheint aber aus einer simplen Sammlung von vorhergehenden Forschungsergebnissen entstanden zu sein. Auch die Fragen, die im zugehörigen Testinstrument gestellt werden, lassen keinen Bezug zu konkreten Fehlvorstellungen erkennen, sondern wirken viel mehr wie ein einfaches Quiz. So wird z.B. nach der Einheit für die Taktfrequenz von Prozessoren gefragt (mit den Auswahlmöglichkeiten GB, Mbps, GHz und Cps [sic]). Diese Frage bezieht sich zwar auf ein Konzept, zu dem diverse Fehlvorstellungen dokumentiert wurden, allerdings dient diese Frage der Ermittlung von Faktenwissen und es handelt sich somit nach der in Kapitel 2 dieser Arbeit gegebenen Definition nicht um eine Fehlvorstellung.

	Topic
1	Loops
2	Branching Structure
3	Bubble Sort Algorithm
4	Use of different types of typical parameters in procedures
5	Functions and Procedures and use of variables in programming
6	Data, Processing and Information
7	Main Memory and Secondary memory
8	Communication Protocols
9	Representation of Data in Computer Language
10	Transformation of Data in Binary Form
11	World Wide Web (www) and Internet
12	Central Processing Unit (CPU)

Tabelle 4.1.: Computer science topics stated as difficult to be understood by students (Ioannou u. Angeli, 2014)

Granularität

Die in den vorgestellten Studien eingeordneten und bewerteten Konzepte haben eine sehr unterschiedliche Granularität und sind oft unterschiedlich gegliedert. Goldman u. a. (2008) ordnen mehreren Konzepten, die von Zendler u. Spannagel (2008) als eigenständige Konzepte bewertet wurden, Oberkategorien zu (z. B. „Algorithms“ mit den Unterpunkten „Algorithm correctness“, „Algorithm analysis“ und „Decidability, Halting Problem“), während andere Konzepte bereits eine Unterkategorie darstellen. Aufgrund weitgehend fehlender Definitionen und Erklärungen ist dieser Umstand schwierig zu beurteilen und macht es auf Basis dieser Informationen auch schwer bis unmöglich, Ergebnisse zu kombinieren. Es wäre generell sehr ertragreich, die in den zuvor vorgestellten Studien (und ebenso in darüber hinausgehenden Studien) zu kombinieren und daraus einen umfangreicheren Katalog zu bilden.

Dieses Problem ist allerdings auch ein unvermeidbares Resultat der grundlegenden Forschungsausrichtung. Während Goldman u. a. sich auf konkrete Kurse an Universitäten beziehen, extrahieren Zendler u. Spannagel die Konzepte aus dem kompletten ACM Kategoriensystem, das eine Fachklassifikation für die Informatik darstellt und kein Schema für die Informatikausbildung ist. Dies führt zwangsläufig dazu, dass einzelne Themengebiete weniger spezifisch behandelt werden können, da dieses Feld deutlich breiter ist.

Letztendlich muss hinterfragt werden, ob ein umfassendes Modell, das die Schwierigkeit informatischer Konzepte hinreichend und allgemeingültig einordnet, überhaupt realisierbar ist. Die Vielfalt an Ausbildungsmöglichkeiten und Schwerpunkten legt nahe, dass dies nicht der Fall ist. Dies stellt allerdings kein Problem dar, solange nachvollziehbar ist, wie entsprechende Konzepte hergeleitet wurden und welchem Kontext sie entnommen wurden. Bei einem Kategoriensystem oder nationalen bzw. internationalen Curricula ist dies beispielsweise durch die Strukturierung und die im Regelfall vorhandenen Beschreibungen gegeben.

Ziel und Zielgruppe der Studien

Das Ziel und die Zielgruppe der zuvor vorgestellten Studien sind unterschiedlich bzw. auch oft nicht definiert. So formulieren Zendler u. Spannagel ihr Ziel als die Ermittlung von zentralen Konzepten für die Entwicklung von Schulcurricula. Das Rating wird allerdings auf Basis eines fachwissenschaftlichen Kategoriensystems von Informatikprofessoren durchgeführt. Durch ihre Definition als „Informatik-Experten“ ist dies einerseits plausibel. Andererseits muss gleichzeitig bedacht werden, dass hier Erfahrungen aus der Schulpraxis nicht oder nur sekundär einfließen.

An dieser Stelle soll darauf verzichtet werden, die einzelnen Zielgruppen und Studien vergleichend aufzulisten. Zweifellos lässt sich aber erkennen, dass vielfach ein schwieriges Spagat zwischen Theorie und Praxis eingegangen wird: zwischen dem, was aus theoretischer Perspektive elementar für ein Fach ist und dem, was es aus praktischer Perspektive ist, bestehen im Allgemeinen große Unterschiede.

4.2.3. Schlussfolgerung

Zusammenfassend lässt sich sagen, dass die in Unterabschnitt 4.2.2 vorgestellten Ergebnisse vielschichtig und nur schwer zu vereinen sein. Offensichtlich können verschiedene Faktoren Studien so sehr beeinflussen, dass unterschiedliche, teilweise sogar konträre Resultate entstehen.

So liegt beispielsweise im Ranking von Zendler u. a. die Logik („logic“) in einem mittleren Cluster (I2/-S). Dieses zeichnet sich in der Bewertung anhand der Kriterien der fundamentalen Ideen durch niedrige Werte beim Vertikalkriterium und Sinnkriterium, sowie hohe Werte beim Zeitkriterium aus. Das bedeutet, dass die Experten der Logik nur eine geringe Vermittelbarkeit auf jeglichem intellektuellen Niveau zusprechen. In der Studie von Goldman u. a. wurde die Schwierigkeit der

vier Unterthemen der Logik („Conditional Statements“, „Boolean algebra, prop logic“, „English to predicate logic“ und „Quantifiers and predicate logic“) mit durchschnittlich 6,43 von 10 Punkten bewertet. Dies entspricht im Vergleich zu den weiteren Konzepten einer Schwierigkeit im mittleren Bereich, deutlich unter den am höchsten bewerteten Konzepten. Auch ein Vergleich der Bewertung der Wichtigkeit in der Studie von Goldman u. a. und dem Gesamtergebnis aus der Studie von Zendler u. a. zeigt, dass in ersterer für die Logik sehr hohe Werte vergeben wurden, während in letzterer der Gesamtscore (d. h. der Mittelwert aller bewerteten Kriterien) im unteren Mittelfeld liegt. Übertragen auf die Forschungsansätze bedeutet dies, dass die Logik einerseits wichtig ist, andererseits aber kein zentrales Konzept ist, das im Informatikunterricht vermittelt werden sollte. Dies steht zwar nicht eindeutig konträr zueinander, scheint aber widersprüchlich zu sein. Im Allgemeinen sollte davon auszugehen sein, dass besonders wichtige Konzepte eines Faches auch Teil des Schulcurriculums sein sollten.

Die Herleitung der zu bewertenden Konzepte ist unterschiedlich. Während diese bei Zendler u. a. durch eine strikt textbasierte Analyse extrahiert wurden, haben Goldman u. a. die Experten explizit nach Themen befragt, die sowohl wichtig als auch schwierig sind. Shinners-Kennedy u. Fincher hingegen haben diese anhand der Kriterien eines Threshold Concepts empirisch erhoben. Die Ergebnisse zeigen einerseits große Übereinstimmungen, andererseits lassen sich auch entscheidende Unterschiede erkennen. So findet das Gebiet der objektorientierten Programmierung mit sechs Unterkategorien bei Goldman u. a. Beachtung und ist bei Shinners-Kennedy u. Fincher eines der beiden als besonders problematisch identifizierten Gebiete, während Zendler u. a. dieses gar nicht betrachtet (wobei anzunehmen ist, dass andere gemessene Konzepte auch verschiedene objektorientierte Konzepte einschließen). Eine Folge der „freien“, ungeleiteten Erhebung von Konzepten scheint zu sein, differenzierte Antworten zu erhalten, die auf den entsprechenden Bereich Bezug nehmen und keine Zusammenfassung darstellen.

Die Granularität (vgl. Abschnitt 4.2.2) führt offensichtlich zu einer starken Beeinflussung der Ergebnisse, indem eine Verfeinerung bzw. Aufteilung von Themengebieten eine deutlich differenziertere Bewertung zulässt. Dies zeigt sich besonders im Vergleich der einzelnen Unterthemen. So wurde bei Goldman u. a. (2008) den drei Unterthemen von „Sets“ Schwierigkeitswerte von 3,9, 5,5 und 9,0 zugeordnet. Ob eine Bewertung des Gesamtthemas dementsprechend einen Durchschnittswert der Einzelbewertungen erhalten hätte oder alternativ von dem niedrigsten bzw. höchsten Rating beeinflusst worden wäre, bleibt unklar.

Ausserdem ergeben sich Unterschiede in der Fragestellung, die auch mit der Zielgruppe in Verbindung stehen. So fragen Goldman u. a. (2008) explizit nach der Wichtigkeit und Schwierigkeit der Konzepte in Einführungskursen. In die Erhebung von Shinners-Kennedy u. Fincher (2013) flossen auch die Erfahrungen von Absolventen ein, die ihren Wissenszuwachs über die gesamte Phase der Ausbildung beschreiben sollten (vgl. Boustedt u. a., 2007). Dass diese Perspektiven die Ergebnisse beeinflussen können bzw. beeinflusst haben, ist anzunehmen.

Abschließend muss die Frage gestellt werden, warum es in der Informatik bisher nur ansatzweise gelungen ist, grundlegende, bedeutsame und besonders schwierige Konzepte zu identifizieren. Der vielfach genannte Grund des jungen Alters des Fachgebiets ist zweifellos nachvollziehbar. Allerdings ist dies nicht nur, wie häufig geschlossen, gleichbedeutend mit dem entsprechenden Zeitraum (im Sinne von „Die Physik wird viel länger gelehrt."), sondern auch mit der damit zusammenhängenden Strukturierung des Faches. Diese ist in der Informatik zum einen sehr uneinheitlich und war zumindest in der Vergangenheit einem starken Wandel unterlegen. Zum anderen scheint die Informatik auch deutlich enger verzahnt zu sein als andere Wissenschaften. Beides wird aus längerfristigen Analysen und Beobachtungen der Forschung in der Informatik deutlich. Pham u. a. (2011) zeigen dies anhand von Graphen, aus denen abzulesen ist, wie z. B. Data Mining (das heute als eigenständige Vorlesung gelehrt wird) aus der Künstlichen Intelligenz hervorging und sich so praktisch in der Struktur der Informatik bewegt hat. Andere Fachgebiete, wie beispielsweise Netzwerke, trennen sich immer mehr von ihrer „Mutterdisziplin" und verbinden sich mit anderen Disziplinen. Auf diese Weise entstehen zwangsläufig neue Zusammenhänge und Abhängigkeiten, die auch in die Lehre einfliessen. Es ist anzunehmen, dass dies in einem Fach mit langer Tradition, z. B. der Physik, deutlich seltener geschieht.

Nichtsdestotrotz ist es verwunderlich, dass Methoden, die in anderen Fächern lang bewährt zur Identifikation entsprechender Konzepte eingesetzt werden (wie z. B. die Erstellung von Concept Inventories), in der Informatik bislang nicht zu befriedigenden Ergebnissen geführt haben. Eine mögliche Begründung ist darin zu sehen, dass oftmals versucht wurde, das komplette Fachgebiet oder zumindest große Teile davon zu analysieren und zu bewerten. Dies führt dementsprechend zu Ergebnissen wie dem zuvor genannten Ansatz von Shinners-Kennedy u. Fincher (2013), die in der Praxis nicht nutzbar sind. Hier wurden schon zu Beginn der Studie Konzepte aus dem gesamten Gebiet der Informatik gesammelt. Dies führt zu Problemen, die auch schon in anderen Fächern auftraten (wie beispielsweise in der Biologie, vgl. z. B. Taylor (2006)). Die einzelnen Fachbereiche liegen zu weit auseinander, als dass tatsächlich zuverlässig übergreifende Threshold Concepts

identifiziert werden könnten. Gleichzeitig besteht die Schwierigkeit, identifizierte Konzepte so zu definieren, dass sie allen Fachbereichen gerecht werden. Dies führt zu Konzepten wie „graphs“ (aus der Studie von Shinners-Kennedy u. Fincher (2013)), deren Funktion und Bedeutung sich über die einzelnen Disziplinen der Informatik hinweg kaum zusammenfassen lässt. Vielfach scheint das Ziel von Forschungsvorhaben schon vorab thematisch so breit und tief angesetzt zu sein, dass dieses realistisch betrachtet nicht zu erreichen ist. Dies bezieht sich weniger auf die theoretische Herleitung, als auf die anschließende empirische Validierung, die mit einem viel zu umfangreichen Instrument erst gar nicht möglich ist. Vergleichbare Ergebnisse aus der Physik beschränken sich zumeist auf enge Teilgebiete, wie z. B. das zuvor vorgestellte „Force Concept Inventory“, das nur das Grundwissen in der Mechanik erfasst. Dies scheint ein sinnvoller Weg zu sein, um zu Ergebnissen zu gelangen, die auch in der Praxis anwendbar sind und nicht aufgrund ihres Umfangs reduziert oder zusammengefasst werden müssen.

4.3. Erhebung von zentralen Konzepten und der Häufigkeit von Fehlvorstellungen

Aus der zuvor gezogenen Schlussfolgerung wird deutlich, dass sich vorhandene Forschungsergebnisse nur schwer adaptieren lassen, um ein eindeutiges und zielgerichtetes Testinstrument für das Wissen über Fehlvorstellungen zu erstellen, das sich speziell auf Themen konzentriert, die sowohl wichtig, als auch herausfordernd für Lernende sind und die Lehrende als besonders „anfällig“ für die Entwicklung von Fehlvorstellungen ansehen. Dies ist insbesondere darauf zurückzuführen, dass manche Ergebnisse thematisch sehr stark eingegrenzt sind, andere hingegen viel zu offen und weitreichend sind. Gleichzeitig unterscheiden sie sich aufgrund unterschiedlicher Methodik, Zielgruppen und Zielsetzungen und können so nur schwerlich kombiniert werden. Hinzu kommt, dass Informationen zu Fehlvorstellungen oft nur beiläufig einfließen, auf einer falschen oder unvollständigen Definition basieren oder manchmal sogar komplett fehlen.

Aus diesem Grund soll im Kontext dieser Arbeit eine Erhebung durchgeführt werden, um auf breiter Basis die zentralen Konzepte der Anfangsphase der Informatikausbildung zu ermitteln und ihre Rolle in Bezug auf Fehlvorstellungen zu messen. Dies soll anhand der Kriterien der fundamentalen Ideen (des Horizontal-, des Vertikal-, des Zeit- und des Sinnkriteriums, vgl. Abschnitt 4.2.1) und einer

Einschätzung der Häufigkeit von Fehlvorstellungen geschehen. Das Hauptziel dieser Vorgehensweise ist es, diese beiden Aspekte in einer Erhebung zu vereinen und damit ein einheitliches und abgeschlossenes Bild zu erlangen. Gleichzeitig soll die Einschränkung auf die Konzepte der Einführungskurse dabei helfen, den Kreis potentieller Konzepte zu verkleinern und damit eine möglichst ausgewogene Zusammenstellung relevanter Konzepte zu erhalten.

Um eine erste Auswahl der zu bewertenden Konzepte zu treffen und den Umfang der Erhebung auf einem akzeptablen Level zu halten, muss vorab eine Eingrenzung erfolgen. Die Adaption eines vorhandenen Klassifizierungsschemas wie z. B. des ACM Computing-Classification-Systems, liegt nahe, birgt jedoch auch die Gefahr, dass dieses Konzepte enthält, die in der Praxis der Informatikausbildung keine Rolle spielen. Die Vorauswahl hat hingegen genau das gegenteilige Ziel. Taxonomien, die exakt diese relevanten Konzepte bestimmen, sind rar und haben oftmals den Nachteil, dass sie regional oder thematisch fokussiert sind. Eine Möglichkeit, dies zu vermeiden, besteht in der Analyse übergeordneter Curricula und Empfehlungen. Zwar sind diese häufig nicht in einer Form, die eine unmittelbare Identifikation von Konzepten erlaubt, aber mit einer Häufigkeitsanalyse können diese identifiziert und nicht relevante bzw. Füllwörter ausgeschlossen werden. Dieses Verfahren wurde z. B. von Zendler u. a. erfolgreich angewandt.

Dieser Ansatz stellt zum einen sicher, dass die zu bewertenden Konzepte grundsätzlich Teil der Informatikausbildung sind. Durch ihre Aufnahme in das Curriculum kann zum anderen auch davon ausgegangen sind, dass sie bereits über einen längeren Zeitrahmen in der Lehre erprobt wurden und somit auch Erfahrungen auf Seiten der Lehrenden bezüglich zugehöriger Fehlvorstellungen bestehen. Als weltweit anerkanntes und genutztes Curriculum bietet sich das ACM/IEEE Curriculum 2013 (ACM/IEEE-CS Joint Task Force on Computing Curricula, 2013) an, das im Folgenden zunächst kurz vorgestellt werden soll.

4.3.1. ACM/IEEE Curriculum 2013

Das ACM/IEEE Curriculum 2013 stellt die aktuellste Ausgabe der etwa in einem 10-Jahres Rhythmus herausgegebenen Empfehlungen für den Entwurf von Curricula für grundständige Studiengänge („undergraduated“) in der Informatik dar. Das Curriculum definiert Inhalte in drei Kategorien: *Core Tier-1*, *Core Tier-2* und *Elective*, die ansteigend in ihrer Relevanz für die Lehre in dieser Phase sind. Themen, die in *Core Tier-1* einsortiert sind, sollten in jedem Informatik Curriculum vorhanden sein, während *Core Tier-2* zu mindestens 80% und *Elective* als Vertiefung

einzelner Themen einbezogen werden sollte. Die Inhalte des *Core Tier-1* enthalten somit die zentralen und verbindlichen Inhalte für die Informatikausbildung.

Die Entscheidung für das ACM/IEEE Curriculum als Grundlage für den Textkorpus basiert auf mehreren Überlegungen. Auch wenn das ACM/IEEE Curriculum eine Empfehlung zum Entwurf von Undergraduate Studiengängen an Universitäten ist, zeigt ein eingehender Vergleich der untersten Stufe (Tier 1) mit deutschen Lehrplänen für die Sekundarstufe 2 große inhaltliche Übereinstimmungen. Der Grund dafür ist vor allem darin zu sehen, dass das Curriculum die Basis für den kompletten Studiengang Informatik darstellt und dabei davon ausgeht, dass die Studierenden keine Vorkenntnisse haben. Durch die Kategorisierung lassen sich zudem weitere Einschränkungen problemlos vornehmen, indem entsprechende fortgeschrittene Themen bzw. Spezialisierungen ausgelassen werden. Ein weiterer Vorteil ist in der internationalen Ausrichtung des ACM/IEEE Curriculums zu sehen. Zwar ist es hinsichtlich des Entstehungsprozesses stark US-amerikanisch geprägt, hat aber in vielen Ländern Einfluss auf die Entwicklung von Curricula gehabt.

Zur Zusammenstellung des Textkorpus wurden die dem Tier-1 zugeordneten Themen extrahiert. Dies geschah durch die Übertragung aller relevanten Inhalte aus dem Originaldokument in ein Textformat.

4.3.2. Textanalyse

Für alle im Folgenden genannten Operationen und Transformationen wurde das tm-Paket für R genutzt. Das zusammengestellte Textmaterial wurde zunächst in Kleinbuchstaben transformiert. Anschließend wurden Stoppworte entfernt. Dabei handelt es sich um Worte, die keine Relevanz für den Inhalt des Textes haben, insbesondere Artikel, Konjunktionen und Präpositionen. Des Weiteren wurden die Interpunktion und Ziffern bzw. Zahlen gelöscht. Der nun vorhandene Textkorpus wurde mit einem Stemming-Algorithmus modifiziert, der die Worte in ihre Grundform zurückführt, um sie einheitlich zählbar zu machen. Nach Durchführung dieser Schritte wurde deutlich, dass eine weitere manuelle Bearbeitung nötig ist. Zwar lassen sich durch diese bewährte Vorgehensweise grundsätzlich zuverlässig Ergebnisse bei der Häufigkeitsanalyse erzielen. Aufgrund der Besonderheiten der Fachbegriffe der Informatik kann dies jedoch auch zu falschen Resultaten der führen. So würde in einem streng automatisierten Vorgehen z. B. der Begriff „data-structures“ als zwei einzelne Worte, d. h. als „data“ und als „structures“, gezählt. Um dies zu

vermeiden, wurden manuell alle zusammengesetzten Fachbegriffe verbunden, um sie somit zunächst als zusammengehörige Begriffe zu behandeln.

Anschließend wurde die Häufigkeitsanalyse durchgeführt. Diese wurde in mehreren Iterationen durchlaufen, da mehrere Begriffe synonym verwandt wurden und einzelne Singular- und Pluralformen nicht korrekt identifiziert wurden. Die Ergebnisse sind in Tabelle 4.2 aufgelistet und in Abbildung 4.2 noch einmal mit einer Wordcloud grafisch veranschaulicht. Dabei werden nur die Konzepte abgebildet, die fünf Mal oder häufiger erwähnt wurden. Die Konzepte „design“ und „programming“ stellen mit jeweils 26 Erwähnungen die häufigsten dar. Dieses Ergebnis ist nicht überraschend, da beide übergeordnete und grundlegende Gebiete der Informatik sind. Dies gilt auch für die darauf folgenden Konzepte „models“ und „software“ mit jeweils 19 Erwähnungen. In den unteren Bereichen finden sich mehrere Begriffe, die als solches nicht gut verständlich sind und einer Erklärung bzw. Definition durch das ACM/IEEE Curriculum bedürfen, um zu gewährleisten, dass diese im weiteren Verlauf der Erhebung korrekt betrachtet werden. Dies trifft z. B. auf „tools“ zu, die sich hier auf Werkzeuge bei der Softwareentwicklung beziehen, die insbesondere bei der Entwicklung und dem Testen genutzt werden.

4.3.3. Implementierung und Durchführung

Die zuvor ermittelten Konzepte (vgl. Tabelle 4.2) wurden in eine Umfrage integriert, die mit Hilfe des Umfragetools „Limesurvey“ implementiert wurde. Sie wurde in fünf einzelne Teile aufgeteilt, in denen die Konzepte jeweils hinsichtlich des Horizontal-, des Vertikal-, des Zeit- und des Sinnkriteriums, sowie die Häufigkeit von Fehlvorstellungen, auf einer fünfstufigen Likert-Skala (von 1 bis 5) bewertet werden sollten. Diese bezog sich bei den Kriterien der Fundamentalen Ideen auf die Aussage, dass das Konzept das Kriterium erfüllt mit der Skala *1 = „Ich stimme zu.“* bis *5 = „Ich stimme nicht zu.“*. Bei der Bewertung von Fehlvorstellungen verlief die Skala von *1 „Fehlvorstellungen treten selten bis nie auf“* bis *5 „Fehlvorstellungen treten sehr häufig auf“*. Die Umfrage gliedert sich somit in zwei einzeln zu betrachtende Komplexe, die die zu messenden Kriterien darstellen: einerseits die Einschätzung des Konzeptes als fundamentale Idee (entsprechend des Ansatzes von Zendler u. Spannagel, 2008), andererseits die Einschätzung der Häufigkeit von Fehlvorstellungen. Ersteres bildet sich durch vier einzelne Kriterien heraus, die zusammen eine Bewertung ergeben, ob es sich bei dem jeweiligen Konzept um eine fundamentale Idee handelt. Letzteres wird durch eine einzelne, eindeutige Bewertung erfragt. Aus

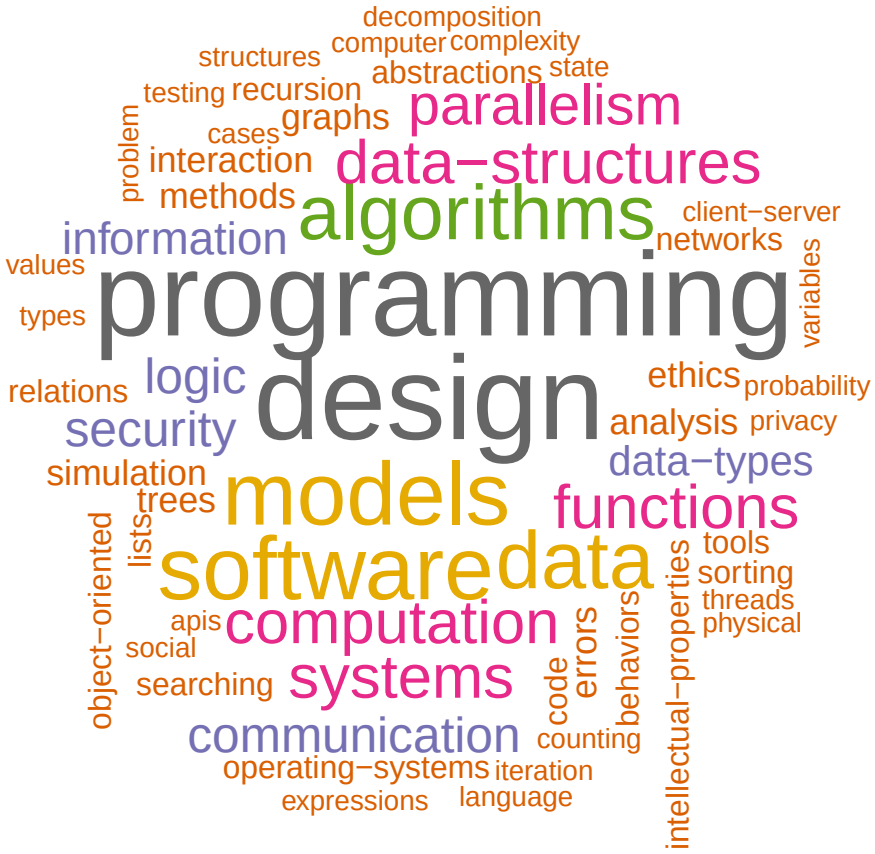


Abbildung 4.2.: Wordcloud der in der Häufigkeitsanalyse identifizierten Konzepte

Anzahl	Konzept	Anzahl	Konzept
26	design	6	graphs
26	programming	6	interaction
19	models	6	methods
19	software	6	simulation
17	data	6	trees
16	algorithms	5	abstractions
12	computation	5	behaviors
12	data-structures	5	code
12	functions	5	intellectual-properties
12	systems	5	lists
11	parallelism	5	networks
9	communication	5	object-oriented
9	logic	5	operating-systems
9	security	5	recursion
8	information	5	relations
7	data-types	5	searching
6	analysis	5	sorting
6	errors	5	tools
6	ethics		

Tabelle 4.2.: Ergebnis der Häufigkeitsanalyse des ACM/IEEE Curriculums ($n >= 5$)

diesem Grund sollen diese Bereiche im Folgenden auch zunächst getrennt voneinander betrachtet und ausgewertet werden und anschließend erst zusammengeführt werden. Für einzelne Konzepte wurde eine kurze Erläuterung angegeben, um ihren Kontext zu definieren und Fehlinterpretationen zu vermeiden. Dies gilt z.B. für „intellectual properties“ („Geistiges Eigentum: Patente, digitale Rechte, Plagiate, etc.“) und „systems“ („Betriebssysteme, Rechnerarchitektur, Speicher“).

Die Erhebung wurde zunächst in einem Pretest mit sieben Informatiklehrkräften an Gymnasien und Gesamtschulen in Nordrhein-Westfalen durchgeführt. Bei der Auswertung zeigte sich schnell, dass die Bewertungen sehr stark voneinander abwichen und trotz der geringen Anzahl der Teilnehmer die gesamte Skala (von Rang 1 bis 5) bei mehreren Konzepten und ihren Kriterien überspannt wurde. Die Standardabweichung war bei der überwiegenden Mehrheit der zu bewertenden Konzepten sehr hoch ($\sigma \geq 2$). Nur vereinzelte Ergebnisse konnten statistische Gütekriterien erfüllen.

Dieses Ergebnis bestätigte sich im Anschluss bei informellen Interviews, die mit drei Teilnehmern des Pretests durchgeführt wurden. In diesen wurde zunächst sichergestellt, dass die Kriterien (bezüglich der fundamentalen Ideen und der Häufigkeit von Fehlvorstellungen) verstanden und richtig angewandt wurden. Im Anschluss wurde nach Begründungen für einzelne Bewertungen gefragt, bei denen sich in der Erhebung eine besonders hohe Standardabweichung feststellen lies. Da die Interviews nur informell durchgeführt wurden, wurden diese nicht aufgenommen, transkribiert und formal ausgewertet. Dieser Schritt stellte lediglich eine Art Orientierung dar, um beurteilen zu können, inwieweit das Setup der Umfrage zu den oben beschriebenen Ergebnissen geführt hat. Die vermutete Fehlerhaftigkeit des Setups bestätigte sich jedoch nicht. Vielmehr wurde deutlich, dass viele Bewertungen stark von subjektiven Einschätzungen und individuellen, teilweise auch einzelnen Erfahrungen geprägt waren. So waren die Bewertungen hinsichtlich möglicher Fehlvorstellungen durch einen Lehrenden, der regelmässig Informatik Leistungskurse unterrichtet hatte, durchgängig niedriger (d. h. er hielt Fehlvorstellungen allgemein für deutlich weniger verbreitet). Auch einzelne persönliche Erfahrungen („Da habe ich mal etwas erlebt ...“) schienen mehrfach als objektives Bewertungskriterium genutzt zu werden.

Aufgrund dieser Problematik wurde ein weiterer Pretest mit vier Professoren an deutschen Universitäten durchgeführt, die Vorlesungen in der Studieneingangsphase in der Informatik gehalten haben. Dieser zeigte deutlich bessere Ergebnisse, da insbesondere die Standardabweichung auf einem deutlich geringeren, akzeptablen Niveau lag. Die Autorin dieser Arbeit hat sich daher dazu entschieden, die Haupterhebung mit dieser Gruppe an Probanden durchzuführen.

Diese Entscheidung bringt mit sich, dass die Bewertung von einem Personenkreis vorgenommen wird, die deutlich geringere Berührungspunkte mit den Lernenden hat. Einführungsveranstaltungen werden häufig für eine große Anzahl an Studierenden konzipiert und durchgeführt. Der Lernerfolg und Lernprozess eines einzelnen kann dabei nicht genau beobachtet werden. Dies bedeutet gleichzeitig, dass Anpassungen an Lehrkonzepten nur sehr langsam erfolgen können, wenn z. B. Klausurergebnisse nicht den Erwartungen entsprachen. Nichtsdestotrotz sollte dieses Wissen und diese Erfahrung auch nicht unterschätzt werden. Ein entsprechendes Setup, in dem die Befragten einen tendenziell theoretischen Hintergrund hatten und zu praktischen Themen befragt wurden, wurde mehrfach in Forschungsansätzen kritisiert, auch wenn es sich oftmals, ähnlich wie in diesem Fall, kaum vermeiden lässt. Die Kritik lässt sich grundsätzlich nachvollziehen und erscheint angebracht, allerdings fehlt ein Beleg, inwieweit ein solches Vorgehen tatsächlich zu schlechteren bzw. sogar falschen

Ergebnissen führt. Im Gegensatz zeigt sich z. B. bei Goldman u. a. (2008), dass sich auf diesem Weg erfolgreiche und überzeugende Studien durchführen lassen.

Zur Umfrage wurden 85 Lehrende an Universitäten (vornehmlich Professoren, aber auch Lehrbeauftragte und wissenschaftliche Mitarbeiter) per E-Mail eingeladen, die in den vergangenen beiden Semestern (Sommersemester 2014 und Wintersemester 2014/2015) eine grundlegende Vorlesung in der Informatik gehalten haben. Als Auswahlgrundlage dienten die ersten 20 Plätze im CHE-Ranking in der Kategorie Lehre. Von den angeschriebenen Lehrenden haben 27 die Umfrage ausgefüllt. Dies entspricht einer Rückläuferquote von ca. 32 %, die ein sehr zufriedenstellendes Ergebnis darstellt.

4.3.4. Zusammenfassung der Ergebnisse

An dieser Stelle sollen nun die Ergebnisse der Studie vorgestellt werden. Zu diesem Zweck werden zunächst die Gesamtergebnisse zusammengefasst, bevor im späteren Teil dieses Abschnittes auch noch auf einzelne Ratings eingegangen wird, um Besonderheiten vorzustellen und zu erläutern.

Die durchschnittliche Bewertung des Horizontal-, des Vertikal-, des Sinn- und des Zeitkriteriums und der Häufigkeit von Fehlvorstellungen ist in Abbildung 4.3 veranschaulicht. Darin werden die einzelnen Kategorien als Spalten und die zu bewertenden Konzepte als Zeilen abgebildet. Die Beschriftung ist auf der unteren bzw. rechten Seite der Matrix zu finden. Der in der Tabelle angegebene Wert entspricht dem arithmetischen Mittel aller Bewertungen. Zusätzlich zu diesem Wert sind die Zellen farbig hinterlegt, um diesen im Vergleich zu den anderen Bewertungen. Diese Farbgebung verläuft von einem dunklen Blau für die niedrigsten Werte zu einem dunklen Rot für die höchsten Werte.

Hinsichtlich der vier Kriterien der fundamentalen Ideen haben *algorithms* ($\bar{x} = 1.18$), *data* ($\bar{x} = 1.36$), *logic* ($\bar{x} = 1.40$) und *information* ($\bar{x} = 1.44$) die niedrigste durchschnittliche Bewertung erhalten. Entsprechend der Forschungsfrage der Erhebung und der Definition von Schwill sind dies somit die Kriterien, bei denen die Bewertenden am eindeutigsten zustimmen, dass diese eine fundamentale Idee sind. Die höchsten Bewertungen erhielten *behaviors* ($\bar{x} = 2.90$), *parallelism* ($\bar{x} = 3.15$) und *object-orientation* ($\bar{x} = 3.32$). Das durchschnittliche Rating für alle Kriterien aller Konzepte ist $\bar{x} = 2.17$. Kein Konzept bekam ein durchschnittliches Rating eines Kriteriums höher als $\bar{x} = 3.59$ für die Kriterien der fundamentalen Ideen (bei

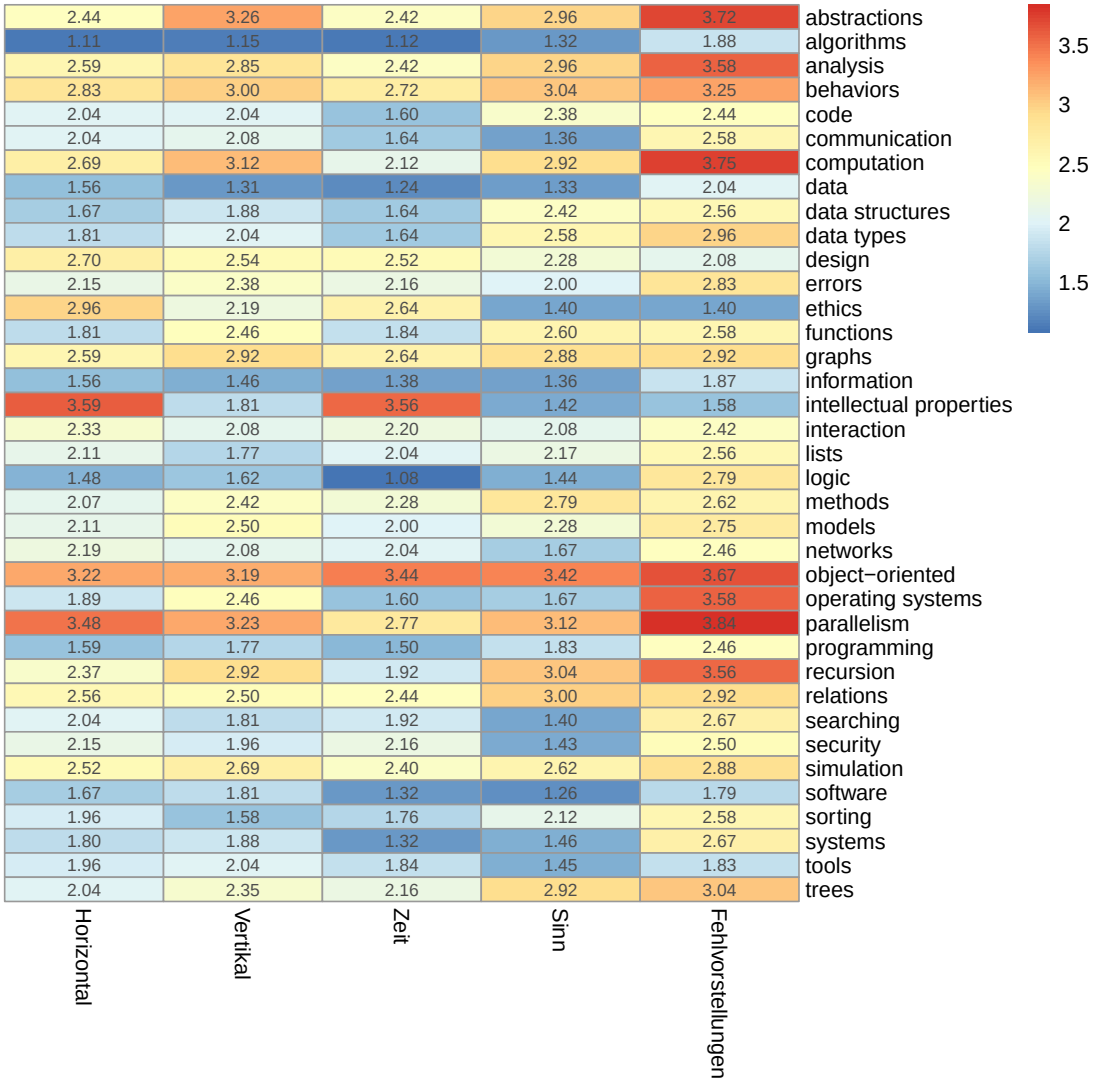


Abbildung 4.3.: Heatmap der Kriterien der fundamentalen Ideen und der Fehlvorstellungen

exakt diesem Wert handelt es sich um das Rating für das Horizontalkriterium von *intellectual properties*).

Zur Messung der Reliabilität wurde die Intra-Class-Correlation (ICC) für alle Ratings (die Kriterien der fundamentalen Ideen und die Häufigkeit von Fehlvorstellungen) berechnet. Mit diesem Verfahren lässt sich messen, inwieweit die Bewertenden in ihren Beurteilungen übereinstimmen (auch bekannt als Interrater-Reliabilität, Shrout u. Fleiss (1979)). Mit einem Reliabilitätskoeefizienten von .93 für die durchschnittlichen Ratings ist das Ergebnis sehr hoch. Für die einzelnen Kriterien beträgt die ICC .91 bis .95, mit dem niedrigsten Ergebnis für das Vertikalkriterium.

Abbildung 4.4 gibt einen Überblick über die Korrelationen der einzelnen Kategorien. Auf der diagonalen Achse (d. h. von der ersten Zeile, erste Zelle, bis zur letzten Zeile, letzte Zelle) sind Histogramme der Durchschnittsbewertungen der einzelnen Kriterien. Hier lässt sich erkennen, dass insbesondere das Horizontal- und das Vertikal-Kriterium nahezu normalverteilt sind. Auf der rechten Seite der Matrix wird die paarweise Korrelation der einzelnen Kriterien angegeben und ihre Signifikanz durch eine Sternwertung verdeutlicht. Das Rating erstreckt sich hier von 0 bis 3 Sternen (höchste Signifikanz). Auf der linken Seite befinden sich Streudiagramme zu den einzelnen Kriterien, die jeweils durch eine LOESS-Glättung veranschaulicht werden.

Hinsichtlich des Kriteriums für die Häufigkeit von Fehlvorstellungen zeigt sich eine starke Korrelation zwischen diesem und dem Vertikalkriterium ($r = 0.75$), sowie dem Sinnkriterium ($r = 0.74$). Das bedeutet, dass durch die Bewertenden eingeschätzt wurde, dass bezüglich eines Konzeptes weniger Fehlvorstellugnen vorhanden sind, wenn es eines dieser beiden Kriterien erfüllt. Im Umkehrschluss sagt dies aus, dass die Häufigkeit für Fehlvorstellungen niedriger ist, wenn das Konzept auf jedem kognitiven Niveau vermittelbar ist oder das Konzept einen Lebensweltbezug hat.

Für das Vertikalkriterium ist dies offensichtlich: wenn ein Konzept auf jedem intellektuellen Level nachvollzogen werden kann, bestehen eventuell auch weniger Fehlvorstellungen darüber, da es schon zu einem sehr frühen Zeitpunkt erlernt und erprobt wurde. Dies ist allerdings insofern erwähnenswert und zu hinterfragen, da Alltagserfahrungen auch gerade bei der Entwicklung von Fehlvorstellungen eine wichtige Rolle spielen (vgl. Abschnitt 3.4.2). Dies führt wiederum zum Sinnkriterium, bei dem auch eine starke Korrelation vorhanden ist. Ein gutes Beispiel für diese Beobachtung ist die Rekursion. Sie hat in ihrer in der Informatik vermittelten Form keinen Alltagsbezug. Dies bestätigt sich auch in der hohen Bewertung von $\bar{x} = 3.04$.

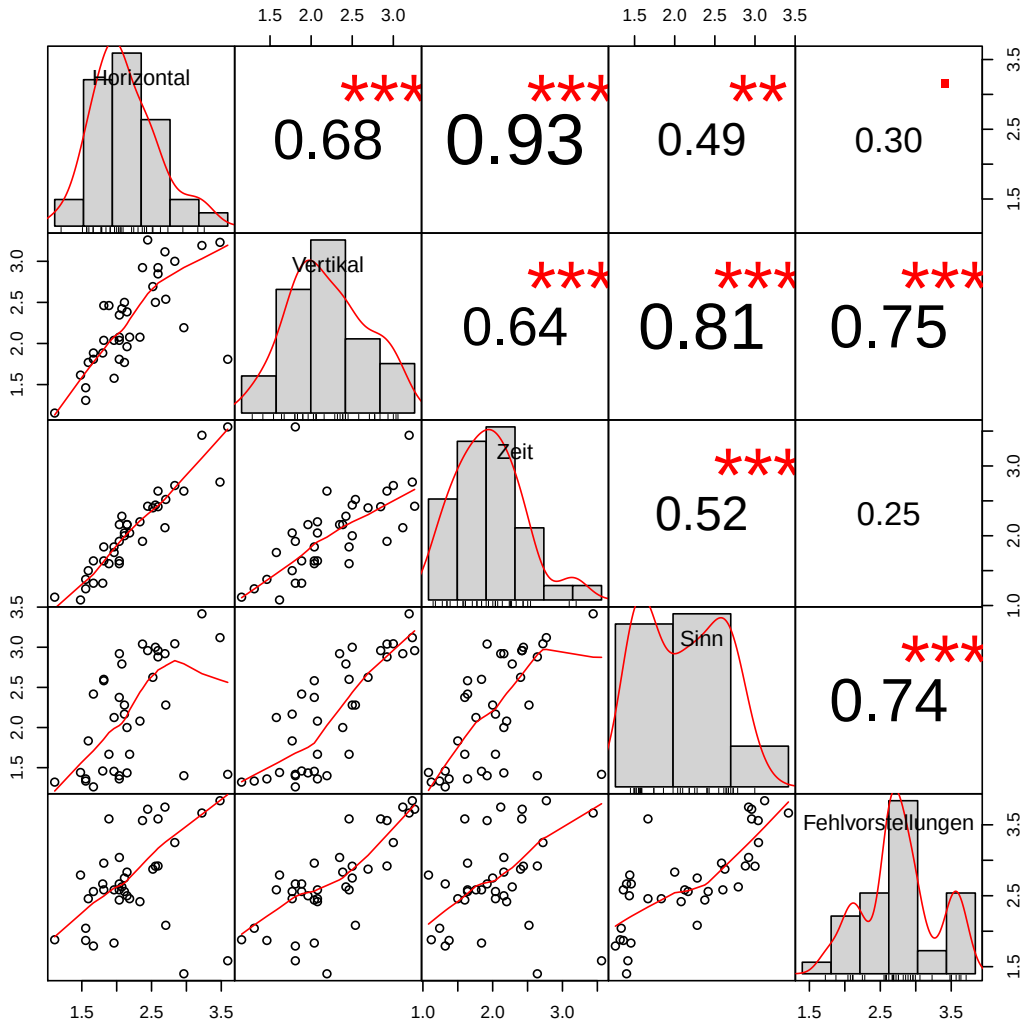


Abbildung 4.4.: Korrelations Matrix der einzelnen Kategorien

Die Häufigkeit von Fehlvorstellungen wurde folgerichtig auch hoch eingeschätzt. Sie lag bei $\bar{x} = 3.56$.

Im Vergleich der Ratings miteinander zeigt sich, dass die individuellen Bewertungen des Sinnkriteriums ($\sigma = 0.68$) und des Kriteriums für die Häufigkeit von Fehlvorstellungen ($\sigma = 0.63$) die höchste Standardabweichung haben, d. h. die Bewertenden waren sich hier besonders uneinig. Die Werte liegen allerdings nicht in einem Bereich, in dem dies besonders auffällig wäre.

Beispiele für Einzelratings

Bei der näheren Betrachtung der Bewertungen zeigen sich mehrere Ergebnisse für individuelle Konzepte, die nicht den Erwartungen entsprechen bzw. die sich nicht eindeutig begründen lassen. Diese sollen beispielhaft im Folgenden vorgestellt werden.

Die Objektorientierung (*object-orientation*) erhielt ein durchschnittliches Gesamtrating von $\bar{x} = 3.32$ bei den Kriterien der fundamentalen Ideen. Das Zeitkriterium wurde hierbei am schlechtesten bewertet ($\bar{x} = 3.44$). Auch wenn die Idee der Objektorientierung schon in den späten 1950er Jahren entstand, gewann das Programmierparadigma erst in den 1990er Jahren stark an Popularität. Die Relevanz der Objektorientierung in der Vergangenheit und insbesondere auch in der Zukunft ist diskutabel und scheint von den Teilnehmern der Studie als dementsprechend niedrig eingeschätzt worden zu sein. Nichtsdestotrotz ist sie heute Thema in der überwiegenden Mehrheit von Einführungskursen der Informatik und ist an vielen Universitäten sogar zur Standardsprache geworden. Trotzdem scheinen viele der Lehrenden in dieser Studie nicht davon überzeugt zu sein.

Die Standardabweichung für die Bewertungen der Objektorientierung ist insgesamt höher ($\sigma = 1.135$) als für andere Konzepte (mean $\sigma = 0.906$, max $\sigma = 1.40$). Das bedeutet, dass die Teilnehmer das Konzept nicht sehr konsistent bewertet haben. Die Gründe für diese Abweichungen können in der Gesamtheit nicht mehr nachvollzogen werden. Häufig scheint aber die objektorientierte Programmierung als solches nicht als bedeutsames Konzept angesehen zu werden. Viel mehr wird sie als Werkzeug für viele andere Konzepte verwendet, auch wenn ein Seiteneffekt davon ist, dass auch die Objektorientierung gelehrt wird. Dies mag einer der Gründe sein, warum die Teilnehmer der Studie den Kriterien nicht zugestimmt haben und dementsprechend die Objektorientierung auf diese Weise auch nicht als fundamentale Idee eingeordnet haben.

Ein weiteres Beispiel ist das Konzept *design* mit einer sehr hohen Bewertung ($\bar{x} = 2.7$) für das Horizontalkriterium. Da dieses Kriterium damit nicht eindeutig erfüllt ist, ist es als Ergebnis dieser Studie auch keine fundamentale Idee. Allerdings ist *design* eine der „Great Principles of Computing“ nach Denning (2003). Es liegt die Vermutung nahe, dass hier eine unterschiedliche Interpretation des Begriffes vorliegt. Denning schließt Software und System Design in dieses Konzept ein und somit auch den Entwurf von Hard- und Software. Als alleinstehender Begriff kann *design* natürlich auch als künstlerische Aktivität (z. B. in Form einer Bildbearbeitung) aufgefasst werden. Auch wenn die in der Umfrage angegebene Definition nahelegt, dass hier „design“ als grundlegendes Konzept der Informatik aufzufassen ist, mag dies ein Grund für die hohe Bewertung sein.

Hinsichtlich der Häufigkeit von Fehlvorstellungen erhielten *ethics* ($\bar{x} = 1.40$) und *intellectual properties* ($\bar{x} = 1.58$) die niedrigsten Bewertungen. In Verbindung mit den Ergebnissen für das Vertikal- und das Sinnkriterium lässt sich interpretieren, dass beide als ein sehr geläufiges und im Alltag relevantes Thema angesehen werden. Dass hier nach Erfahrung der Studienteilnehmer nur sehr wenige Fehlvorstellungen vorhanden sind, ist ohne Blick auf die übrigen Bewertungen jedoch durchaus interessant. Es hat sich vielfach bestätigt, dass auch Studierende der Informatik sich der ethischen Grundlagen, die in der Disziplin existieren, nicht bewusst sind. Nicht ohne Grund gab es in den letzen Jahrzehnten eine Vielzahl an Forderungen, auch ethische Themen in der Informatik Lehre zu vermitteln (z.B. Miller, 1988; Ben-Ari, 2001). Das niedrige Rating zur Häufigkeit von Fehlvorstellungen mag darin begründet sein, dass diese Themen fachwissenschaftlich nicht anspruchsvoll sind. Das dies gleichzeitig auch bedeutet, dass weniger Fehlvorstellungen vorhanden sind, ist diskussionswürdig.

Die höchsten Bewertungen bei der Häufigkeit von Fehlvorstellungen erhielten *computation* ($\bar{x} = 3.75$) und *parallelism* ($\bar{x} = 3.84$). Hierbei lassen sich Gemeinsamkeiten mit den Threshold Concepts feststellen, die in der Studie von Shinners-Kennedy u. Fincher (2013) durch die befragten Experten als solche genannt wurden. Ein weiteres der genannten Threshold Concepts ist *Abstractions*, das auch in der vorliegenden Studie das dritthöchste Rating bei der Häufigkeit von Fehlvorstellungen bekam ($\bar{x} = 3.72$). Genauso liegt die *object-orientation* auf dem vierten Rang ($\bar{x} = 3.67$), die sich durch *objects* und *classes* als Threshold Concepts wiederfindet. Ein Gegenbeispiel für diese Gemeinsamkeiten ist das Konzept *algorithms*, dem eine niedrige Bewertung in dieser Studie gegeben wurde ($\bar{x} = 1.88$), das aber durch Shinners-Kennedy u. Fincher auch als Threshold Concept genannt wird.

Die niedrigsten Bewertungen für die individuellen Kriterien (für die Kriterien der fundamentalen Ideen und die Häufigkeit von Fehlvorstellungen) wurden für das Zeitkriterium vergeben ($\bar{x} = 2.04$). Das bedeutet, dass die Konzepte überwiegend als relevant in der Vergangenheit, als auch in der Zukunft der Informatik angesehen werden. Dies ist kein überraschendes Ergebnis, da sie durch ihr Vorkommen in den grundlegenden Veranstaltungen des ACM/IEEE Curriculums bereits als fundamental und bewährt klassifiziert werden können.

Die Verteilung der Ratings für alle Kriterien zeigt, dass mehr als 61 % der abgegebenen Bewertungen bei 1 und 2 lagen (d. h. die Studienteilnehmer haben zugestimmt, dass das jeweilige Konzept das Kriterium erfüllt bzw. dass für dieses Konzept wenige Fehlvorstellungen entstehen) und nur 14 % mit 4 und 5 (was wiederum einer Ablehnung des Kriteriums entspricht). Zusätzlich zu den durchschnittlichen Bewertungen der Konzepte, die zuvor vorgestellt wurden, zeigt auch dies, dass für die überwiegende Mehrheit eine Zustimmung vorhanden ist.

Die zuvor dargestellten Beobachtungen und Interpretationen müssen bei der Analyse und Weiterverwendung der Ergebnisse beachtet werden. Ein Median aller Mittelwerte der Konzepte von 2.17 zeigt, dass ein Konzept, das sich im Ranking im unteren Bereich befindet, nicht zwangsläufig keine fundamentale Idee ist, sondern lediglich in Relation zu den anderen Konzepten schlecht bewertet wurde. Diese Verteilung der Bewertungen war jedoch durch die Wahl der Quelle vorhersehbar. Die grundlegenden Kurse des ACM/IEEE Curriculum enthalten nur Inhalte, die sich in Einführungsveranstaltungen bewährt haben und stellen somit eine bereits „gefilterte“ Menge dar. Durch die zusätzliche Beschränkung auf die grundlegenden Inhalte (aus Tier-1) und die durchgeführte Häufigkeitsanalyse wird dieser Effekt zusätzlich verstärkt. Darauf zurückzuführen ist auch der Umstand, dass es nahezu keine extremen Bewertungen im oberen Bereich für einzelne Konzepte gab (d. h. eine Bewertung eines Kriteriums der fundamentalen Ideen > 4). Zusammenfassend stellt dies allerdings kein Problem dar, sondern sollte lediglich im Forschungsprozess nicht außer Acht gelassen werden.

Um die nun vorhandenen Daten weiterverarbeiten zu können, müssen diese tiefgehender analysiert werden. Aufgrund der Konstellation der einzelnen Kriterien und der Bedeutung entsprechender Bewertungen erscheint es nicht angemessen, lediglich Durchschnittswerte für jedes Konzept zu bilden und auf diesem Wege die höchstbewerteten Konzepte zu ermitteln. Dies würde dazu führen, dass niedrige Bewertungen durch hohe ausgeglichen werden und ein Konzept somit zu einer fundamentalen Idee wird, obwohl es eines der Kriterien nicht erfüllt und damit

nicht der Definition von Schwill entspricht. Dies muss in jedem Fall vermieden werden, indem alle Kriterien individuell betrachtet werden.

Gleichzeitig ist naheliegend, die Kriterien der fundamentalen Ideen und die Bewertung der Häufigkeit von Fehlvorstellungen getrennt voneinander zu betrachten, da erstere gemeinsam ein Kriterium bilden, indem sie bei einer positiven Bewertung aller Kriterien ein Konzept zu einer fundamentalen Idee machen.

Aus diesen Gründen sollen die Konzepte mit Hilfe einer Clusteranalyse in mehrere homogene Gruppen eingeteilt werden, um spezifische Gemeinsamkeiten herauszustellen. Schon in der zuvor erfolgten Betrachtung einzelner Ratings hat sich gezeigt, dass sich hier verschiedene Muster wiederfinden. Es liegt nahe, diese entsprechend zu sortieren und zu interpretieren. Dies kann auch dabei helfen, Zusammenhänge zwischen den einzelnen Kriterien bzw. insbesondere zwischen den Kriterien der fundamentalen Ideen und der Häufigkeit für Fehlvorstellungen zu identifizieren.

4.3.5. Clusteranalyse

Auf Basis der erhobenen Daten wird nun eine Clusteranalyse durchgeführt. Dabei stellt sich zunächst die Frage, ob eine Clusterung anhand aller fünf gemessenen Merkmale oder zunächst lediglich anhand der Kriterien der fundamentalen Ideen geschehen sollte.

Wie bereits zuvor beschrieben, sind die einzelnen Konzepte nicht disjunkt und lassen sich teilweise hierarchisch anordnen. Die Bewertungen zeigen hier allerdings, dass sich diese Tatsache nicht so auf die Einschätzung der Häufigkeit von Fehlvorstellungen auswirkt, dass Konzepte und ihre Unterkonzepte identische Bewertungen erhalten hätten. Ganz im Gegensatz zeigen sich hier sehr wechselhafte Verbindungen, die allerdings auch darin begründet sind, dass eine eindeutige hierarchische Zuordnung nicht möglich ist.

Dies legt nahe, dass hier eine differenzierte Auswertung unter Beachtung der jeweiligen Zusammenhänge und Verknüpfungen vorgenommen werden sollte. Des Weiteren besteht durch die Korrelation zwischen dem Kriterium für die Häufigkeit von Fehlvorstellungen und dem Vertikalkriterium, sowie dem Sinnkriterium die Gefahr, dass dieses Kriterium in der Analyse überrepräsentiert ist. Aus diesem Grund wird zunächst eine Clusteranalyse für die Kriterien der fundamentalen Ideen durchgeführt.

Zur Durchführung der Clusteranalyse wurde der k-Means-Algorithmus genutzt. k-Means ist ein Verfahren, bei dem zunächst die Anzahl der Cluster (k) festgelegt werden muss. Im Gegensatz zu hierarchischen Verfahren, bei denen die Menge der zu clusternden Elemente immer weiter aufgeteilt wird bis jeder Cluster nur noch ein Element enthält, ergibt sich diese nicht als Ergebnis der Analyse bzw. kann nach Durchführung des Clusterings ausgewählt werden. Ziel von k-means ist es, einen Datensatz so zu unterteilen, dass die $WCSS$ (within-cluster sum of squares, Summe der quadrierten Abweichungen innerhalb der Cluster) minimal ist. Dies geschieht, indem zunächst k Punkte als Anfangszentren festgelegt werden. Danach werden in mehreren Iterationen die Punkte den am nächsten gelegenen Zentren zugeordnet und die Zentren werden neu berechnet, bis sich die Zentren nicht mehr ändern. Das Ergebnis wird durch die Wahl der Anfangszentren beeinflusst. Aus diesem Grund wird die Berechnung mit k-means mit n randomisiert gewählten Startzentren durchgeführt und das Ergebnis mit der niedrigsten $WCSS$ ausgewählt (in der vorliegenden Arbeit $n = 100$).

Die generelle Problemstellung von k-Means kann dazu genutzt werden, die Anzahl der Cluster zu bestimmen, indem dieser Wert für unterschiedliche k berechnet und verglichen wird. Zu beachten ist hier, dass die $WCSS$ mit ansteigendem k sinkt, da der Abstand der einzelnen Elemente zu den Zentren kleiner wird, desto mehr von diesen existieren. Diese sinkt jedoch im Regelfall sprunghaft, wenn die Anzahl der Cluster unter einen bestimmten Wert sinkt (auch als „Ellbow“ bezeichnet). Die bei der Auswertung der für diese Arbeit erhobenen Daten sind in Abbildung 4.5 zu sehen. Hier zeigt sich, dass die WCSS bis k = 4 stark sinkt und nach k = 7 deutlich abflacht.

Dies wird auch durch den Dunn Index bestätigt, der ein Maß zur Validierung von Clustern ist und dazu das Verhältnis zwischen dem kleinsten Abstand zwischen zwei separierten Clusterpaaren und dem maximalen Durchmesser eines Clusters berechnet (s. Abbildung 4.6). Dieser steigt zwischen $k = 6$ und $k = 7$ deutlich an. Auf Basis dieser Maße wird die nun folgende Clusteranalyse mit 7 Clustern durchgeführt.

Abbildung 4.7 zeigt die Verteilung der einzelnen Cluster mit Hilfe eines zweidimensionalen Plots. Um die Werte auf diese Weise abzubilden, wurde eine Hauptkomponentenanalyse durchgeführt. Die Achsen entsprechen somit den beiden ermittelten Variablen. Die Varianz liegt hier bei 94,37 %, was bedeutet, dass sich ein hoher Anteil der in den Ergebnissen der Clusteranalyse enthaltenen Informationen auf diese Weise abbilden lässt. Die Darstellung zeigt besonders deutlich, dass zwei Cluster mit jeweils zwei Konzepten abseits der übrigen fünf Cluster liegen, während sich diese

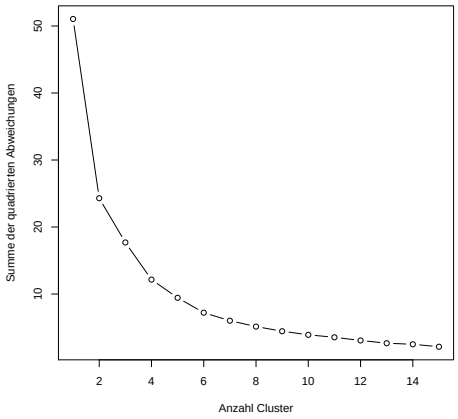


Abbildung 4.5.: Summe der quadr. Abweichungen innerhalb der Cluster für die fundamentalen Ideen

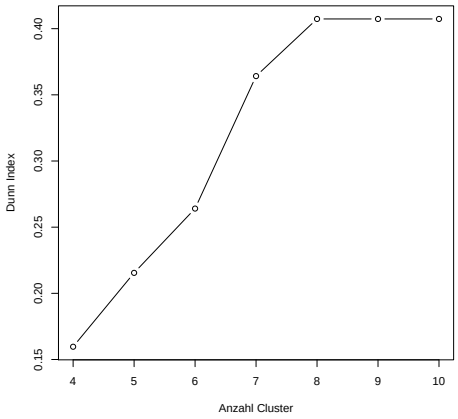


Abbildung 4.6.: Veranschaulichung des Dunn-Index nach Anzahl der Cluster

im rechten oberen Bereich sammeln. Durch die Darstellung des 4-dimensionalen als 2-dimensionaler Plot lässt sich dies nicht direkt auf die gemessenen Kriterien übertragen. Entsprechende Zusammenhänge werden aber aus der nun folgenden Beschreibung der Cluster deutlich.

Stabilität und Validität der Cluster

Zur Überprüfung der Stabilität der Cluster wurden zwei zusätzliche Clusteranalysen mit einer unterschiedlichen Anzahl von Datensätzen ($n = 15$ und $n = 20$) durchgeführt und der Rand-Index (Rand, 1971) berechnet. Dieser gibt einen Wert für die Ähnlichkeit von zwei Clustern an und liegt bereits für $n = 15$ bei einem überzufälligen Wert von 0.59. Für $n = 20$ beträgt er 0.68. Die Ergebnisse der Clusteranalyse sind somit stabil.

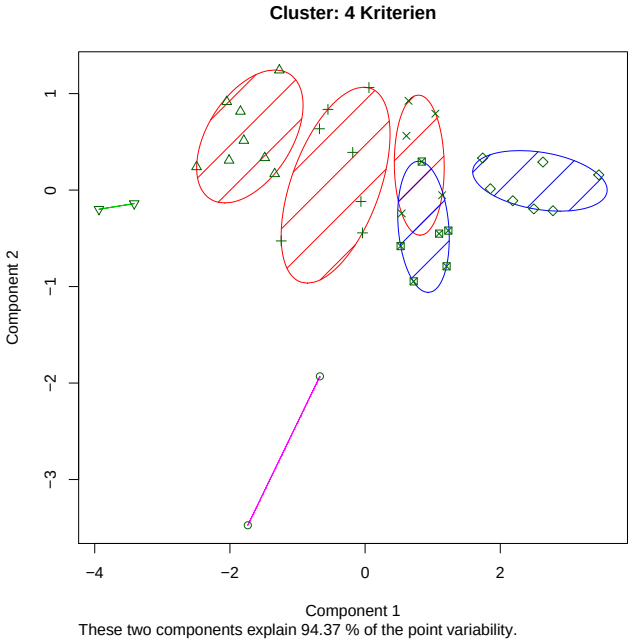


Abbildung 4.7.: Visualisierung der Clusteranalyse für $k = 7$ für die Kriterien der Fundamentalen Ideen

Cluster	Horiz.	Vert.	Zeit	Sinn	Fehlv.	Konzepte
T1	1.54	1.57	1.28	1.43	2.21	algorithms, data, information, logic, programming, software, systems
T2	1.92	1.86	1.74	2.33	2.60	code, data structures, data types, lists, sorting
T3	2.04	2.07	1.87	1.50	2.61	communication, networks, operating systems, searching, security, tools
M1	2.17	2.39	2.17	2.42	2.62	design, errors, functions, interaction, methods, models, trees
M2	2.57	2.91	2.38	2.93	3.32	abstractions, analysis, behaviors, computation, graphs, recursion, relations, simulation
O1	3.28	2.00	3.10	1.41	1.49	ethics, intellectual properties
O2	3.35	3.21	3.10	3.27	3.75	object-oriented, parallelism

Tabelle 4.4.: Verteilung der Cluster und der Ratings für die fundamentalen Ideen und die Häufigkeit von Fehlvorstellungen

Beschreibung der Cluster und Interpretation

In Tabelle 4.4 befindet sich eine Auflistung der Cluster, ihrer durchschnittlichen Bewertungen für die einzelnen Kriterien und die darin enthaltenen Konzepte. Die Namensgebung erfolgte nach dem Prinzip einer Rangordnung. Es wurden drei Cluster mit besonders niedrigen Bewertungen bei den Kriterien der fundamentalen Ideen identifiziert (im Folgenden bezeichnet als $T1$, $T2$ und $T3$), zwei mit mittleren Werten ($M1$ und $M2$) und zwei ausserhalb der üblichen Schemata ($O1$ und $O2$).

Die in Cluster $T1$ enthaltenen sieben Konzepte haben die niedrigsten Werte beim Horizontal-, Vertikal-, Zeit- und Sinnkriterium (1.54, 1.57, 1.28, 1.43). Sie wurden als diejenigen Konzepte bewertet, die die Kriterien der fundamentalen Ideen am Besten erfüllen. Es sind: *algorithms*, *data*, *information*, *logic*, *programming*, *software* und *systems*. Aus dieser Zusammenstellung wird deutlich, dass es sich um sehr grundlegende Konzepte handelt, die im ACM/IEEE Curriculum eigene Kategorien darstellen (Algorithms and Complexity, Information Management, Software Engineering und System Fundamentals). Des Weiteren werden sie häufig auch an Universitäten im Rahmen eigenständiger Vorlesungen unterrichtet. Abgesehen

von *logic* wurden alle Konzeptes dieses Clusters auch in der Studie von Zendler u. Spannagel (2008) zustimmend bewertet.

Cluster $T2$ schließt sich dem ersten Cluster an, hat allerdings bei allen Kriterien eine höhere Bewertung (1.92, 1.86, 1.74, 2.33). Darin enthalten sind die Konzepte *code*, *data structures*, *data types*, *lists* und *sorting*. Besonders deutlich wird die höhere Bewertung beim Sinnkriterium, das als einziges Kriterium in diesem Cluster einen durchschnittliche Wert über 2 bekam.

$T3$ ist der dritte Cluster im T-Set und enthält *communication*, *networks*, *operating systems*, *searching*, *security* und *tools*. Dieser Cluster hat, insbesondere im Gegensatz zum vorhergehenden Cluster, ein signifikant niedrigeres Rating beim Sinnkriterium (2.04, 2.07, 1.87, 1.50). Die Bewertenden sprechen den Konzepten einen hohen Alltagsbezug zu. Dies ist insofern nachvollziehbar, da alle Konzepte als solche auch ausserhalb der Fachwissenschaft bekannt und auch Computernutzern, die kein vertieftes Wissen in der Informatik haben, vertraut sind. Dies gilt insbesondere für Betriebssysteme und die Suche. Dass dies nicht zwangsläufig ein Indiz dafür ist, dass sie problemlos zu lehren bzw. zu lernen sind, zeigt sich in der Bewertung der Häufigkeit von Fehlvorstellungen: diese ist mit $\bar{x} = 2.61$ höher als bei den vorhergehenden Clustern T1 und T2. Ganz im Gegenteil lässt sich auch wieder die Verbindung zur Theorie ziehen, dass durch die Verknüpfung von Wissen, das im Alltag erworben wurde, mit neuem Wissen Fehlvorstellungen entstehen können (vgl. Abschnitt 4.2). Diese Beobachtung wiederholt sich allerdings nicht in den übrigen Clustern.

Nach dem T-Set folgen zwei Cluster mit mittleren Bewertungen, die als $M1$ und $M2$ bezeichnet werden. Beide haben für die Kriterien der fundamentalen Ideen kein durchschnittliches Rating unter $\bar{x} = 2$ und kein Rating über $\bar{x} = 3$. Markant für Cluster $M2$ ist ein hoher Wert bei der Häufigkeit von Fehlvorstellungen, der, abgesehen von Cluster $O2$, den höchsten Durchschnittswert darstellt.

$M1$ besteht aus *design*, *errors*, *functions*, *interaction*, *methods*, *models* und *trees*. Die Bewertungen für alle Kriterien der fundamentalen Ideen sind sehr konsistent (2.17, 2.39, 2.17, 2.42). In Cluster $M2$ befinden sich die Konzepte *abstractions*, *analysis*, *behaviors*, *computation*, *graphs*, *recursion*, *relations* und *simulation*. Er hat, im direkten Vergleich mit $M1$, höhere Bewertungen für alle Kriterien, insbesondere für das Vertikalkriterium und das Sinnkriterium (2.57, 2.91, 2.38, 2.93). Zudem wurde die Häufigkeit für Fehlvorstellungen deutlich höher eingeschätzt ($\bar{x} = 3.32$). Auffallend ist, dass im Gegensatz zu anderen Clustern deutlich mehr methodische bzw. prozedurale Konzepte enthalten sind.

Der sechste und siebte Cluster fallen insbesondere durch ihre geringe Größe auf. Beide enthalten nur jeweils zwei Konzepte. Ihre Sonderstellung wird auch aus Abbildung 4.7 deutlich, wo sie sich jeweils in den Randbereichen befinden. Sie werden im Folgenden einzeln beschrieben und interpretiert.

Cluster $O1$ zeigt eine starke Differenz zwischen dem Horizontal- und Zeitkriterium, sowie dem Vertikal- und Sinnkriterium (3.28, 3.10, 2.00, 1.41). Ein besonders niedriger Wert wurde dem letztgenannten zugeordnet, der auch der niedrigste Wert für dieses Kriterium in allen Clustern ist. Aufgrund der beiden enthaltenen Konzepte *ethics* und *intellectual properties* ist dies nachvollziehbar: durch die immer größer werdende Verbreitung von Informatiksystemen wurden diese Themen vielfach in den Medien thematisiert, um Computernutzer aufzuklären bzw. auf entsprechende Risiken oder Gefahren hinzuweisen, indem z. B. auf die Wahl sicherer Passwörter oder der Schutz persönlicher Daten eingegangen wird. Ein Grundverständnis ist somit schon auf einem sehr niedrigen Niveau möglich. Gleichzeitig ist damit auch eine Alltagsrelevanz gegeben.

Cluster $O2$ hat hingegen gleichmässig hohe Werte bei allen vier Kriterien (3.35, 3.21, 3.10, 3.27). Es handelt sich dabei um die höchsten Bewertungen in den Einzelkategorien (wobei das Zeitkriterium äquivalent zu Cluster $O1$ ist). Interessant ist hier, dass *parallelism* sich in der hierarchischen Weiterführung der Masterideen nach Schubert u. Schwill (2011) befindet und somit aus der Sicht der Autoren eine fundamentale Idee darstellt. Es ist auffällig, dass dieses Konzept in der Studie sehr hohe Bewertungen erhielt und mit einem Durchschnittswert von $\bar{x} = 3{,}15$ auch die zweithöchste Durchschnittsbewertung hat. Es ist damit weder absolut, noch in Relation zu den anderen Konzepten, als fundamentale Idee eingeschätzt worden. Die Vermutung, dass die Konzepte hier aus einer wissenschaftlichen Perspektive beurteilt wurden und somit z. B. das Horizontalkriterium besonders hoch bewertet wurde, kann ausgeschlossen werden, da insbesondere auch das Vertikalkriterium, das bei einer solchen Interpretation nicht beeinflusst werden sollte, entsprechende Bewertungen erhalten hat. Wodurch diese verursacht wurden, lässt sich im Nachgang aufgrund der anonymen Befragung leider nicht mehr nachvollziehen und übersteigt zudem den Umfang dieser Arbeit. Nichtsdestotrotz ist dies eine interessante anschliessende Forschungsfrage, auf die in Kapitel 7 weiter eingegangen werden soll.

Die Bewertungen der einzelnen Kriterien in Relation zueinander wird noch einmal in Abbildung 4.8 veranschaulicht. Jede Graphik präsentiert dabei die Bewertungen der Konzepte anhand von zwei Kriterien. Die jeweiligen Beschriftungen der Achsen sind den Angaben in den Zeilen bzw. Spalten zu entnehmen. So bildet die Zelle

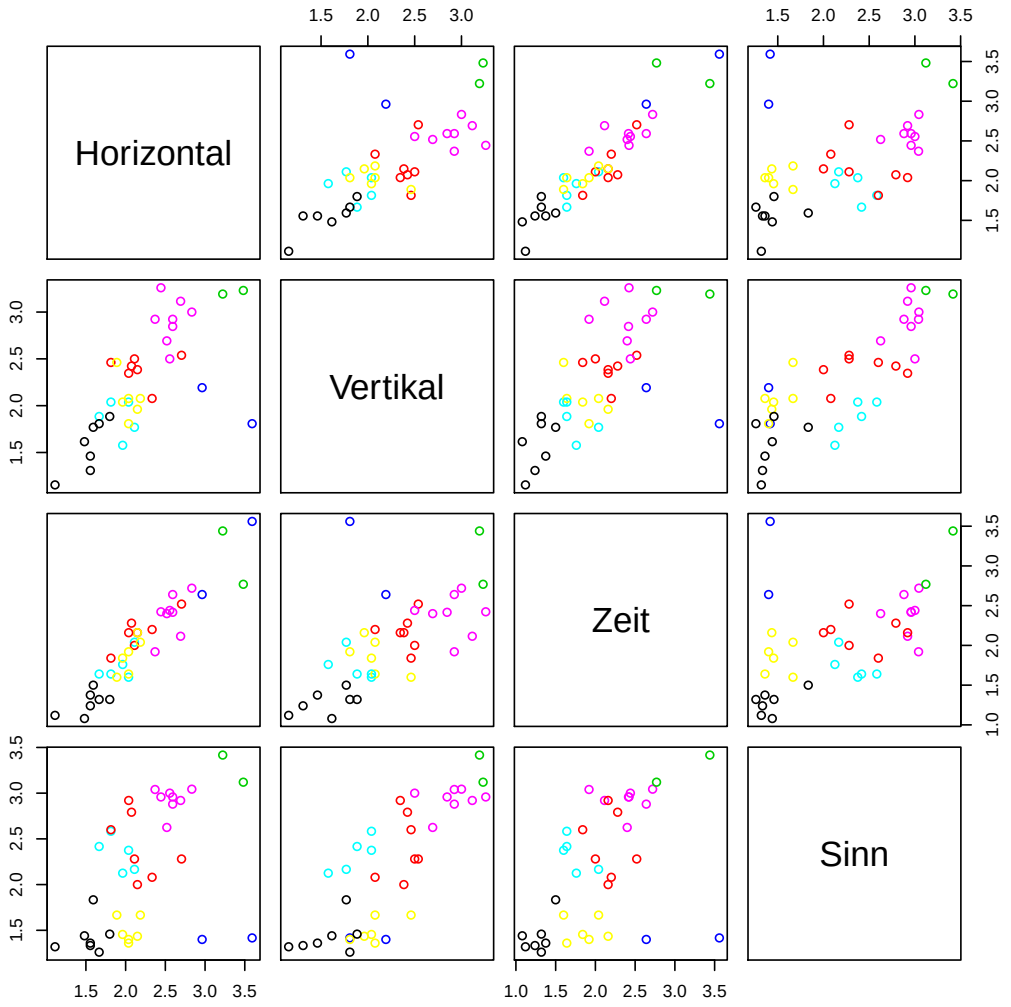


Abbildung 4.8.: Matrix zur Veranschaulichung der Bewertung der Kriterien und der einzelnen Cluster

in der ersten Reihe, zweite Spalte die Bewertungen des Vertikalkriteriums auf der x-Achse und die Bewertungen des Horizontalkriteriums auf der y-Achse ab. Die Punkte, die die Bewertung auf den Skalen angeben, sind farblich den sieben Clustern zugeordnet, d. h. jeder Cluster hat eine individuelle Farbe. Hier lassen sich die zuvor beschriebenen Verteilungen noch einmal graphisch nachvollziehen.

Kriterien für Fundamentale Ideen und Fehlvorstellungen

Bisher fand lediglich eine Clusterbildung anhand der Ratings der Kriterien der fundamentalen Ideen statt. Als logische Fortführung der zuvor durchgeführten Schritte sollte nun auch die Bewertung der Häufigkeit von Fehlvorstellungen in einer Clusteranalyse betrachtet werden. Dieses Vorgehen wurde jedoch nach einer ersten exemplarischen Analyse verworfen. Die Entscheidung ist zum einen darin begründet, dass durch die zuvor vorgestellten Analyse wertvolle Ergebnisse gewonnen werden konnten, die ohne Einschränkung im nächsten Schritt dieser Arbeit, der Erstellung von Testitems, verwendet werden können. Zum anderen wird dies auch klar durch statistische Variablen unterstützt, die verdeutlichen, dass sich die Ergebnisse durch eine breitere Analyse verschlechtern.

Der Vergleich der Clusteranalyse, die anhand der vier Kriterien der fundamentalen Ideen und der Bewertung der Häufigkeit von Fehlvorstellungen durchgeführt wurde, zeigt, dass die Werte für die Validität bei der Clusterung mit vier Kriterien deutlich höher sind. Konkret beträgt der Dunn Index hier 0.3642, während er bei fünf Kriterien bei 0.234 liegt.

Grund dafür ist vor allem die Verschleierung der ursprünglichen Kriterien für die Relevanz der Konzepte (d. h. die Kriterien der fundamentalen Ideen) durch die Hinzunahme des Kriteriums für Fehlvorstellugnen, die dazu führt, dass die Cluster uneinheitlicher und auch kleiner sind. So zeigt sich für Cluster $T1$, der die niedrigsten Werte für vier Kriterien hatte, dass dieser bei Einbeziehung des fünften Kriteriums in drei einzelne Cluster aufgesplittet wird, da die Häufigkeit von Fehlvorstellungen deutlich uneinheitlicher bewertet wurde. Die Cluster verlieren somit ihre klar erkennbaren Charakteristiken. Gleiches gilt auch für die Häufigkeit von Fehlvorstellungen der Konzepte, da diese sich nicht mehr individuell nachvollziehen lassen. Dies ist ein häufiges Problem, wenn die Anzahl der Merkmale bei einer Clusteranalyse steigt: Wenn die Zahl der Cluster nicht ebenso steigt, sinkt die Validität innerhalb der Cluster. Aus diesem Grund sollte vermieden werden, zu viele Kriterien in die Clusteranalyse miteinzubeziehen, wenn sich diese auch

Konzept	Mean	Fehlv.	Konzept	Mean	Fehlv.
parallelism	3.15	3.84	communication	1.78	2.58
computation	2.71	3.75	functions	2.18	2.58
abstractions	2.77	3.72	sorting	1.86	2.58
object-oriented	3.32	3.67	data structures	1.90	2.56
analysis	2.70	3.58	lists	2.02	2.56
operating systems	1.90	3.58	security	1.93	2.50
recursion	2.56	3.56	networks	1.99	2.46
behaviors	2.90	3.25	programming	1.67	2.46
trees	2.37	3.04	code	2.01	2.44
data types	2.02	2.96	interaction	2.17	2.42
graphs	2.76	2.92	design	2.51	2.08
relations	2.62	2.92	data	1.36	2.04
simulation	2.56	2.88	algorithms	1.18	1.88
errors	2.17	2.83	information	1.44	1.87
logic	1.40	2.79	tools	1.82	1.83
models	2.22	2.75	software	1.51	1.79
searching	1.79	2.67	intellectual properties	2.59	1.58
systems	1.62	2.67	ethics	2.30	1.40
methods	2.39	2.62			

Tabelle 4.5.: Bewertung der Häufigkeit von Fehlvorstellungen (in absteigender Ordnung)

einer Clusterung mit weniger Kriterien zuordnen lässt (vgl. z. B. Bortz u. Döring, 2006).

4.3.6. Bewertung der Häufigkeit von Fehlvorstellungen

Schon zuvor wurde auf die Bewertung der Häufigkeit von Fehlvorstellungen einzelner Konzepte eingegangen. An dieser Stelle soll dies noch einmal im Detail geschehen, um die Betrachtung fortzuführen und abzuschließen. Tabelle 4.5 gibt einen Überblick über alle Bewertungen, wobei die Konzepte absteigend geordnet sind. Dazu ist die durchschnittliche Bewertung für die Häufigkeit von Fehlvorstellungen angegeben (Spalte 3). Die Werte spannen sich von 3.84 bis 1.40, mit der Mehrheit der Konzepte zwischen 2 und 3. Während eine Vielzahl der Bewertungen nachvollziehbar ist, gibt es einige Ausreißer, deren Ergebnis die Autorin dieser Arbeit vor der

Auswertung höher eingeschätzt hatte. Dies gilt z. B. für das Konzept *algorithms*, das eine durchschnittliche Bewertung von 1.88 erhielt. Dies ist insofern überraschend, da Algorithmen viele Aspekte beinhalten, die als besondere Herausforderungen für Anfänger in der Informatik gelten und entsprechend dokumentiert wurden. Bemerkenswert ist zudem, dass die fünf Konzepte mit den höchsten Bewertungen bei der Häufigkeit von Fehlvorstellungen jeweils durchschnittliche Bewertungen der Kriterien der fundamentalen Ideen über 2.70 haben. Das bedeutet, dass sie von den Teilnehmern nicht bzw. nur teilweise als fundamentale Ideen angesehen werden.

Cl.	Konzept	Fehlv.	Cluster	Konzept	Fehlv.
$T1$	algorithms	1.88	$M1$	design	2.08
	data	2.04		errors	2.83
	information	1.86		functions	2.58
	logic	2.79		interaction	2.42
	programming	2.46		methods	2.63
	software	1.79		models	2.75
	systems	2.67		trees	3.04
$T2$	code	2.44	$M2$	abstractions	3.72
	data structures	2.56		analysis	3.58
	data types	2.96		behaviors	3.25
	lists	2.56		computation	3.75
	sorting	2.58		graphs	2.92
$T3$	communication	2.58		recursion	3.56
	networks	2.46		relations	2.92
	operating systems	3.58		simulation	2.88
	searching	2.67	$O1$	ethics	1.40
	security	2.50		intellectual properties	1.58
	tools	1.83	$O2$	object-oriented	3.67
				parallelism	3.84

Tabelle 4.6.: Bewertung der Häufigkeit von Fehlvorstellungen zu den einzelnen Konzepten geordnet nach Clustereinteilung

In Tabelle 4.6 sind den jeweiligen Clustern die Bewertungen für die Häufigkeit von Fehlvorstellungen zugeordnet. Daraus ist zu erkennen, dass diese in den Clustern $T1$ bis $M2$ steigt. Die äußeren Cluster $O1$ und $O2$ haben auch hier extreme Werte. Konkret ist dies mit der niedrigsten Bewertung aller Konzepte von 1.40 für *ethics* in $O1$ bzw. der höchsten von 3.84 für *parallelism* in $O2$.

4.4. Auswahl der im Testinstrument zu messenden Konzepte

4.4.1. Vorüberlegungen

Um die Auswertung der Erhebung abzuschließen und in den nächsten Schritt des Forschungsvorhabens überzugehen, muss nun entschieden werden, welche Konzepte in der folgenden Studie zur Messung des Wissens über Fehlvorstellungen behandelt werden sollen. Die zu erfüllende Bedingung ist, dass diese einerseits als für die grundlegende Informatikausbildung relevant angesehen werden, aber gleichzeitig auch eine Lernhürde darstellen, bzw. die Wahrscheinlichkeit für die Entwicklung von Fehlvorstellungen zu diesem Konzept hoch ist.

Eine rein von den Bewertungen der Häufigkeit von Fehlvorstellungen geleitete Herangehensweise ist nicht zu empfehlen. So haben die drei mit der höchsten Häufigkeit bewerteten Konzepte (*parallelism*, *computation* und *abstractions*) eine niedrige Bewertung bei den Kriterien der fundamentalen Ideen und befinden sich damit in den mittleren und äußeren Clustern $M2$ und $O2$. An dieser Stelle müssen zunächst die oberen Cluster betrachtet werden, um so eine Auswahl der zentralen, wichtigsten Konzepte zu erhalten. Erst daraus kann die Identifikation besonders herausfordernder Konzepte erfolgen.

Gleichzeitig muss bei diesem Vorgehen aber beachtet werden, dass schon bei der Auswahl der in der Erhebung zu bewertenden Konzepte deutlich wurde, dass diese nicht trennscharf sind und teilweise Überbegriffe darstellen, denen diverse andere Konzepte zugeordnet werden können. Dies führt dazu, dass sich einzelne Konzepte in einen anderen Cluster transferieren lassen, indem man sie ihrer Überkategorie zuordnet. Dies trifft z.B. für „sorting“ und „searching“ zu (Cluster $T2$ und $T3$), die den Algorithmen in Cluster $T1$ zugeordnet werden können. Gleiches gilt für *data structures* und *data types*, die unter *data* zusammengefasst werden können. Zudem existieren zwischen den Konzepten Querverbindungen, die ebenso bedacht werden müssen. Das bedeutet, dass sich die Konzepte nicht 1:1 einander zuordnen lassen, sondern teilweise mehrfach zugeordnet werden können oder zwangsläufig bei der Anwendung eines Konzepts (insbesondere im Rahmen einer Aufgabenstellung im Testinstrument) ein Teil von diesem sind.

4.4.2. Umsetzung

In Cluster $T1$ befinden sich die Konzepte *algorithms*, *data*, *information*, *logic*, *programming*, *software* und *systems*. Diese haben Bewertungen der Häufigkeit von Fehlvorstellungen zwischen 1.79 (software) und 2.79 (logic). Zu dem etwas niedriger bewerteten Cluster $T2$ gehören die Konzepte *code*, *data structures*, *data types*, *lists* und *sorting* mit einer Häufigkeit von Fehlvorstellungen zwischen 2.44 (code) und 2.96 (data types), die somit höher als für den Cluster $T1$ geschätzt wurde. Interessant ist, dass sich alle Konzepte in $T2$ jeweils Konzepten aus Cluster $T1$ zuordnen lassen. Sie stellen damit ein spezifischeres Konzept dar, welches offensichtlich aufgrund dessen auch als anfälliger für Fehlvorstellungen eingeschätzt wird.

Die Mehrheit der Konzepte lässt sich nun einander zuordnen. Eine Ausnahme stellt *software* dar: sie besteht einerseits aus *algorithms* und *programming*, ist aber auch selbst Teil von *systems*. Dem Konzept *data* lassen sich insgesamt drei Unterkonzepte aus Cluster $T2$ zuordnen: *data structures*, *data types* und *lists*. Von diesen hat *data types* die höchste Einstufung bei der Häufigkeit von Fehlvorstellungen (2.96). Zwischen den ausgewählten Konzepten bestehen zahlreiche Querverbindungen und Zusammenhänge, so dass sich diese nur schwerlich abbilden lassen. Sie sollen aber durch die Erstellung der Items im folgenden Kapitel verdeutlicht werden.

Konzept T1	Konzepte T2	wird behandelt in Item:
algorithms	sorting	Item 1 und 2
data	data structures, data types, lists	Item 2, 3 und 4
information		Item 6
logic		Item 5
programming	code	Item 1, 6
software		Item 1, 6
systems		Item 7

Tabelle 4.8.: Überblick über die Zuordnung der Konzepte in Cluster 1 und 2 und ihre Repräsentation in den Testitems

5. Entwicklung von Testitems für die Informatik

5.1. Einleitung

Im Folgenden soll der Entwicklungsprozess der Items beschrieben werden, die im Rahmen dieser Arbeit erstellt wurden und die anschließend in der Hauptstudie eingesetzt werden (s. Kapitel 6). Dazu soll zu jedem im vergangenen Kapitel identifizierten und ausgewählten Konzept zunächst die Herleitung der grundlegenden Thematik erfolgen und konkrete Ergebnisse aus den Forschungsgebieten reflektiert und analysiert werden. Im Anschluss wird das erstellte Item vorgestellt. Dies wird ergänzt durch eine Musterlösung, sowie die Betrachtung und Einschätzung der zu erwartenden Ergebnisse.

An dieser Stelle erfolgt die Rückführung der ursprünglich aus dem ACM/IEEE Curriculum extrahierten Konzepte in ihren Kontext, um ihnen darauf aufbauend bekannte Fehlvorstellungen zuzuordnen bzw. diese vorzustellen, falls dies zuvor in Kapitel 3 noch nicht geschehen ist. Dies soll weniger als Definition dienen, sondern vielmehr als thematischer Umriss, um die hinter den einzelnen Konzepten stehenden Gebiete einzugrenzen.

5.2. Testtheoretische Kriterien und Testkonstruktion

Die Gestaltung von Testitems muss unter Beachtung spezifischer testtheoretischer Kriterien geschehen. In diesem Abschnitt soll daher zuallererst eine Entscheidung getroffen und begründet werden, wie die Testitems konzipiert werden sollen.

Eine sehr wichtige Rolle nimmt dabei die Wahl des Itemformates ein (vgl. z. B. Moosbrugger u. Kelava, 2011, S. 38). Ein offenes Format, das eine Frage vorgibt, die der Proband frei beantworten kann, hat im Allgemeinen den Vorteil, dass es besonders in wenig bekannten Themengebieten die Möglichkeit bietet, neue und bislang unbekannte Antworten zuzulassen und damit evtl. sogar das Forschungsfeld zu erweitern. Im Themenkomplex der vorliegenden Arbeit könnte dies z. B. eine unerwartete Interpretation einer beschriebenen Fehlvorstellung sein. Aufgrund der Tatsache, dass Fehlvorstellungen in der Informatik wenig dokumentiert sind, ist dies ein erstrebenswertes Nebenergebnis. Dies führt allerdings gleichermaßen zum Problem, dass diese Antworten nicht ohne Schwierigkeiten zu bewerten sind, da insbesondere im zuvor beschriebenen Fall nicht eindeutig ist, ob die Antwort als richtig oder falsch anzusehen ist.

Die Problematik von geschlossenen Fragen ergibt sich bereits aus der vorhergehenden Beschreibung. Das Spektrum der Antwortoptionen muss so definiert sein, dass es eine möglichst große Anzahl an Antwortmöglichkeiten abdeckt, gleichzeitig aber auch eindeutig richtige und falsche Antworten vorgibt. Eine schlechte Zusammenstellung kann dazu führen, dass den Testteilnehmern bei der Beantwortung geholfen wird, indem offensichtlich richtige oder eindeutig falsche Antwortmöglichkeiten angeboten werden (Härtig, 2014). Zudem besteht bei einem Multiple-Choice-Format das Problem, dass Antworten lediglich die Erfahrungen des Aufgabenstellers bzw. entsprechende Forschungsergebnisse beinhalten. In einem nicht umfassend erforschten Gebiet kann dies hinderlich sein. Gleichzeitig hat ein offenes Format einen motivierenden Effekt auf die Befragten, da diese die Möglichkeit bekommen, ihre eigenen Erfahrungen zu teilen und diese nicht auf ein vorgegebenes Schema reduzieren müssen (Porst, 2009).

Aus diesem Grund soll in den folgenden Abschnitten zum Entwurf der Items versucht werden, das Format möglichst ausgewogen zu wählen, so dass je nach Ziel oder Ausrichtung der Frage ein möglichst optimales Ergebnis erzielt werden kann. Dieses gilt insbesondere dafür, dass es eine zuverlässige Messung des Wissens des Befragten ermöglichen, aber auch Optionen für Feedback bzw. eine zukünftige Erweiterung und Veränderung der Fragen offen halten soll.

Ein weiterer zu beachtender Aspekt ist, dass die Testitem aus fachlicher Perspektive von einer möglichst hohen Anzahl an Teilnehmern zu beantworten sein sollten. Das bedeutet in diesem Fall, dass keine Programmiersprache und kein Programmcode verwendet wird, der nicht für alle Probanden verständlich ist. Ebenso sollten möglichst keine Konzepte eingesetzt werden, die nicht Teil der behandelten Fehlvorstellung sind. Dass dies zu unerwünschten Ergebnissen führt, hat sich bereits

im Vortest (vgl. nachfolgender Unterabschnitt 5.2.1) bestätigt, in dem ein Teil der Testteilnehmer nicht in der Lage war, Fehler in einem Klassendiagramm zu identifizieren, da die Notation nicht bekannt war (Ohrndorf u. Schubert, 2013).

5.2.1. Entwicklung und Erprobung der Testitems

Die im folgenden vorgestellten Items wurden in mehreren Iterationen erstellt, mit Testpersonen erprobt und überarbeitet. Dadurch konnten diese immer weiter verbessert werden und bei der Bearbeitung aufgetretene Probleme behoben werden. Dazu zählen insbesondere Verständnisprobleme bei der Aufgabenstellung oder Schwierigkeiten bei der Bearbeitung. In den individuellen Beschreibungen der Items werden hierfür einzelne Beispiele genannt.

5.3. Algorithmen

Das erste identifizierte Konzept ist das Thema Algorithmen. Algorithmen sind im Allgemeinen definiert als Verarbeitungsvorschrift zur Lösung eines Problems und stellen damit eines der elementaren Themen der Informatik und Mathematik dar. Seit Mitte der 70er Jahre, als sich die Informatikausbildung an den Universitäten und Schulen etablierte, stellte die Formulierung und Programmierung von Algorithmen einen zentralen Kern des Unterrichts dar. Dies spiegelt sich in Schwills Definition der Algorithmisierung als Masteridee der Fundamentalen Ideen wider (Schwill, 1993). Die Einordnung macht deutlich, dass dieser Idee weitere Ideen untergeordnet sind, darunter auch andere zur Implementierung in dieser Umfrage ausgewählte Konzepte. Aus diesem Grund sollen hier keine Fehlvorstellungen zu bestimmen Typen von Algorithmen (z. B. Sortier- oder Suchalgorithmen) oder Komponenten von Algorithmen behandelt werden, sondern grundlegende Eigenschaften und Definitionen.

Gal-Ezer u. Zur (2003) beschäftigen sich mit Fehlvorstellungen bei der Einschätzung der Effizienz und Komplexität von Algorithmen. Dabei ermittelten sie die folgenden Fehlvorstellungen:

- Je kürzer ein Programm ist (hinsichtlich der Anzahl der Zeilen), desto effizienter ist es.
- Je weniger Variablen verwendet werden, desto effizienter ist das Programm.

- Zwei Programme mit identischen Statements haben die gleiche Effizienz (auch wenn die Reihenfolge unterschiedlich ist).
- Zwei Programme, die die selbe Aufgabe bearbeiten, haben die gleiche Effizienz.

Die ersten beiden Vorstellungen basieren dabei auf der simplen Annahme „mehr von A, mehr von B“ (more of A, more of B). Die dritte und vierte auf „gleiches A, gleiches B“ (same A, same B). Beide stellen ein häufiges Denkmuster bei Fehlvorstellungen in den Naturwissenschaften und der Mathematik dar (Gal-Ezer u. Zur, 2003, S. 244).

Özdener (2008) hat das Testinstrument von Gal-Ezer u. Zur von 242 Studierenden in der Oberstufe von technisch ausgerichteten Schulen und Universitäten eingesetzt. Die Fragen wurden allerdings von einem Multiple-Choice zu einem offenen Format geändert. Die zuvor an dritter Stelle genannte Fehlvorstellung („Zwei Programme mit identischen Statements haben die gleiche Effizienz.“) konnte in dieser Studie lediglich bei 1 % der Studierenden festgestellt werden. Abgesehen davon konnten die ermittelten Fehlvorstellungen bestätigt werden.

Das dieser Fehlvorstellung zugrundeliegende Wissen hat hinsichtlich seiner Messbarkeit den Vorteil, dass es unabhängig von der jeweiligen gelernten Programmiersprache ist. Um die Effizienz eines Algorithmus festzustellen, kann dieser auch in Pseudocode oder sogar in natürlicher Sprache geschrieben sein. Dieses Thema bietet sich somit aufgrund der in Abschnitt 5.2 genannten Kriterien für das hier zu entwickelnde Testitem an. Auch hinsichtlich der in Kapitel 2 genannten Eigenschaften von Fehlvorstellungen scheint diese geeignet, da sie, falls sie nicht konkret Teil einer Aufgabenstellung sind, stabil bestehen kann.

5.3.1. Item 1 - Algorithmen

Die beiden folgenden Algorithmen ermitteln das Produkt zweier Zahlen ohne Multiplikation und geben dieses aus. In einer Klausuraufgabe sollen die Schüler nun ermitteln, welcher Algorithmus effizienter hinsichtlich seiner Laufzeit ist.

```
1 mult = 0
2
3 x = input("Enter number
      1:")
4 y = input("Enter number
      2:")
5
6 for x in range(1,x+1):
7         mult = mult + y
8
9 print(mult)
```

Listing 5.1: Algorithmus 1

```
1  mult = 0
2  x = input("Enter number
       1:")
3  y = input("Enter number
       2:")
4
5
6  if x < y:
7          limit = x
8          value = y
9  else:
10         limit = y
11         value = x
12
13 for x in range(1,limit
     +1):
14          mult = mult +
              value
15
16 print(mult)
```

Listing 5.2: Algorithmus 2

Aufgabe - Algorithmen

1. Welcher Algorithmus ist effizienter?
   - ☐ Algorithmus 1
   - ☐ Algorithmus 2
2. Nennen Sie mindestens eine Fehlvorstellung, die zu einer falschen Beantwortung dieser Frage führen könnte.

Musterlösung

1. Algorithmus 2 ist effizienter. Während Algorithmus 1 stets die Eingabe für die erste Zahl (Variable x) als Anzahl der Durchläufe der for-Schleife setzt, ermittelt Algorithmus 2 zunächst, welche der eingegebenen Zahlen kleiner ist und wählt die entsprechende Zahl für die Anzahl der Schleifendurchläufe aus. Wenn somit y kleiner als x ist, wird die for-Schleife entsprechend seltener aufgerufen und der Algorithmus ist somit effizienter.
2. Mögliche Gründe für eine falsche Einschätzung:
   - Der Programmcode von Algorithmus 2 ist länger.
   - Algorithmus 2 nutzt mehr Variablen (zusätzlich *loop* und *value*).
   - Algorithmus 2 hat eine zusätzliche if-else-Verzweigung.

Vorbewertung des Items

Das Item verlangt, dass der Befragte zunächst selbst die Aufgabe löst, bevor er eine Einschätzung abgeben kann, welche Fehlvorstellung zu einer falschen Antwort führen könnte. Dies bedeutet, dass nur solche Antworten bei der Auswertung berücksichtigt werden dürfen, die zuvor eine richtige Lösung ausgewählt haben. Die Möglichkeiten für erkennbare Fehlvorstellungen scheinen durch die Literatur bereits weitgehend eingegrenzt zu sein, eine Erweiterung ist jedoch nicht ausgeschlossen.

Interessant könnten hier aber auch die Antworten der Testteilnehmer sein, die zuvor die Aufgabe falsch gelöst hatten. Dabei besteht die Möglichkeit, dass diese exakt die Merkmale nennen, die als Fehlvorstellung interpretiert werden können.

Davon abgesehen sollte das Item leicht zu beantworten sein, wenn die fachwissenschaftliche Frage richtig beantwortet wurde. Die Unterschiede zwischen dem effizienteren und dem weniger effizienten Algorithmus sind dann offensichtlich und problemlos zu identifizieren. Aus diesen Unterschieden sollten sich entsprechend die Fehlvorstellungen erkennen lassen.

5.4. Sortieren

Das Thema *Sortieren* ist eine aus dem Alltag bekannte Tätigkeit, die sich in vielfältiger Art und Weise in der Informatik wiederfindet. Sie wird daher in einfacher Form (d. h. mit simplen Algorithmen) schon früh in der Informatikausbildung angewendet. So enthält *Computer Science Unplugged* (Bell u. a., 1996) eine Aktivität, bei der spielerisch der Selection-Sort- und Quicksort-Algorithmus vermittelt wird, indem die Lernenden unterschiedlich gefüllte Behälter wiegen und nach ihrem Gewicht sortieren.

Simon u. a. (2006) haben mehr als 400 Studierende nach ihrem Vorgehen bei der Konzeption von Sortieralgorithmen befragt. Sie sollten in natürlicher Sprache beschreiben, wie sie eine Liste mit zehn Zahlen aufsteigend sortieren. Die Mehrheit der Testteilnehmer (63 %) verfolgte dabei einen Ansatz, der die Zahlen als String behandelt, also als eine Verkettung von Ziffern. Die Sortierung erfolgte dabei durch das Zählen und Anordnen der Ziffern. Dies ist der aus der Grundschule bekannten Sicht auf Zahlen ähnlich: die Schüler lernen, dass Zahlen aus Ziffern bestehen und die Zahl größer ist, wenn sie aus mehr Ziffern besteht. Bei dieser Studie sollte neben der korrekten Antwort auch der Einfluss einer Einführungsveranstaltung zur Informatik gemessen werden. Hierzu wurden Studierende vor Beginn des Studiums und nach 10 Wochen eines Java-Kurses (CS1) befragt. Bemerkenswert ist, dass zwar der Anteil der „String“-Antworten geringer wurde, aber 33 % der Befragten in der ersten Erhebung eine bessere Beschreibung gaben als in der zweiten. Nur 14 % verbesserten sich in der zweiten Befragung. Somit waren nach dem Besuch der Vorlesung weniger Studierende in der Lage, einen Algorithmus richtig zu formulieren, als vor Beginn der Veranstaltung. Die Autoren sehen den Grund hierfür insbesondere in der Verwendung von Fachsprache. Diese wurde im zweiten Durchlauf deutlich häufiger eingebracht, aber durch fehlendes bzw. lückenhaftes Wissen auch oft falsch verwendet. In einer Fortsetzung dieser Studie befragten Chen u. a. (2007) Studierende nach ihrem Vorgehen, Datumsangaben zu sortieren. Die Ergebnisse waren der zuvor genannten Erhebung sehr ähnlich und bestätigten diese.

Diese Ergebnisse machen deutlich, dass Lernende, auch wenn sie keine oder nur geringe Programmierkenntnisse haben, in der Lage sind, korrekte Algorithmen zu beschreiben. Sie können somit das grundsätzliche Problem lösen und nutzen dazu auch die in der Informatik verwendeten Kontrollstrukturen. Was ihnen allerdings fehlt, ist der korrekte und exakte Umgang mit informatikspezifischen Konstrukten. Keiner der Teilnehmer, der noch keine Programmiererfahrung hatte, nutzte den

Begriff „while“, viele jedoch „repeat“ und „until“. Simon u. a. (2006) vermuten daher, dass es Anfängern deutlich leichter fällt, eine *do-while* Schleife anstatt einer *while-do* Schleife zu verstehen. In jedem Fall machen diese Ergebnisse deutlich, dass Lernende auch zu Anfang der Informatikausbildung mentale Modelle von Konzepten der Informatik haben. Es ist daher wichtig, dass der Lehrende diese erkennt, um sie zum einen nutzen und zum anderen auch um verhindern zu können, dass sie sich zu Fehlvorstellungen entwickeln.

Die im folgenden Item behandelte Fehlvorstellung bezüglich der Behandlung des Datentyps Integer als String ist insofern charakteristisch, als dass sie auf einer grundlegenden Vorstellung beruht, aber als solche aber nicht offensichtlich ist. Diese wird, wie auch Simon u. a. (2006) beschreibt, erst durch die Anwendung bzw. sogar erst durch die Beschreibung der Anwendung, deutlich.

5.4.1. Item 2 - Algorithmen, Sortieren, Datentypen

Aufgabe - Sortieren

Ein Student soll einen Algorithmus beschreiben, der die folgenden 10 Zahlen aufsteigend nach ihrer Größe sortiert:
20, 341, 4, 38, 93, 120, 92, 250, 171, 48

Student: „Ich teile die Zahlen erst einmal nach ihrer Größe auf, also einmal die mit einer Ziffer, dann die mit zwei Ziffern, dann die mit drei. In der ersten Gruppe ist nur eine Zahl, also ist das schon einmal die kleinste. Dann teile ich die zweite Gruppe in weitere Gruppen auf, indem ich die erste Ziffer anschaue und vergleiche. So gehe ich dann weiter vor, bis alle Zahlen angeordnet sind.“

Ist dieses Vorgehen richtig? Welche Fehlvorstellung könnte hier vorliegen?

Musterlösung

Beispiel:
Grundsätzlich wird hier ein Bucketsort Algorithmus beschrieben. Der Student vermischt allerdings die verschiedenen Datentypen. Zunächst erfolgt eine Sortierung anhand der Anzahl der Zeichen (d. h. die Zahlen werden als String interpretiert). Anschließend werden die Zahlen aber innerhalb der verschiedenen Gruppen nach ihrer Größe sortiert. Dies erscheint insbesondere inkonsistent, weil offenbar nicht die Möglichkeit in Betracht gezogen wird, dass mehrstellige Zahlen verglichen werden müssen, die nach dem ersten Durchlaufen der Sortierung noch nicht geordnet sind.

Vorbewertung des Items

Durch die wenig konkrete Aufgabenstellung und die offene Fragestellung werden hier sehr vielseitige Antworten gegeben werden. Die Identifizierung des Bucketsort Algorithmus sollte in jedem Fall möglich sein, ist aber keine Bedingung für eine korrekte Antwort. Hier steht nicht der tatsächliche Sortieralgorithmus im Mittelpunkt, sondern der Umgang mit den Datentypen und die Formulierung des Algorithmus, sowie seiner Bestandteile. Der jeweilige Sortieralgorithmus könnte somit problemlos mit einem anderen Sortieralgorithmus ersetzt werden, um den Umgang mit den Datentypen darzustellen. Der diffuse Umgang mit Zahlen, bei dem zudem auch keine klare Abbruchbedingung definiert ist, ist in jedem Fall eine Bedingung für eine richtige Antwort, da dies ein wichtiger Aspekt bei möglichen Fehlvorstellungen ist. Ähnlich wie bei dem vorhergehenden Item ist es auch hier möglich, dass neue Antworten, die bei der Konzeption nicht eingeplant waren, genannt werden.

5.5. Daten - Datentypen und Datenstrukturen

Der sehr undifferenzierte Begriff *data* hängt sehr stark mit *Information* zusammen und wird in diesem Kontext auch im ACM/IEEE Curriculum verwendet. Ein weiteres Anwendungsgebiet ist die Darstellung und Modellierung von Daten, in diesem Fall mit Hilfe von Datenstrukturen und Datentypen. Aufgrund der höheren Bewertung bei der Häufigkeit von Fehlvorstellungen soll sich dieses Testitem auf die Datentypen konzentrieren.

Tew u. a. (2005) untersuchten das Wissen von Studierenden verschiedener Studienrichtungen vor und nach der Belegung eines CS1-Kurses. Basierend auf Ansätzen von Lister u. a. (2004) wurde ein Testinstrument mit Multiple-Choice-Items verwendet. Dabei handelte es sich zum einen um Tracing-Aufgaben, bei denen die Teilnehmer den Ablauf des Programmcodes verfolgen und anschließend die richtige Antwort geben mussten. Zum anderen waren dies Aufgaben, bei denen ein Teil des Programmcodes vorgegeben wurde, der von den Testteilnehmern vervollständigt werden musste. Getestet wurden sieben grundlegende Themen und sechs fortgeschrittene. Ein Beispiel für ein Tracing-Item ist das Nachverfolgen von verschiedenen Änderungen an Arrays. So wurde z. B. in einem Schritt das dritte Element eines Arrays an die zweite Position eines anderen Arrays übertragen und das richtige Ergebnis sollte ausgewählt werden. Einer der vorgegebenen Distraktoren, der von mehr als 50 % der Ingenieure unter den Testteilnehmern gewählt wurde, ist die falsche Zählweise der Elemente. Diese gingen davon aus, dass gemäß der im Alltag verwendeten Zählweise das erste Element den Index 1 hat. Dementsprechend verschoben sich alle Operationen um ein Element nach links.

Forišek u. Steinová (2012) beschäftigen sich mit Metaphern und Analogien und erwähnen die aus ihrer Sicht fragwürdige Analogie der Warteschlange im Supermarkt für die Datenstruktur *queue* (im Deutschen auch als Warteschlange bezeichnet). Dieser Vergleich kann missverstanden werden, indem die Warteschlange als dynamisch wahrgenommen wird. Zum einen rücken die übrigen Wartenden auf, sobald ein Element (d. h. ein Kunde) die Schlange verlässt. Zum anderen verteilen sich die Elemente auch, bilden neue Schlangen oder einzelne Elemente verlassen die Schlange während sie warten. Forišek u. Steinová nennen aus ihrer Perspektive deutlich geeignetere Analogien, wie z. B. das Warten in einem Raum, in dem Wartenummern ausgegeben werden. Dabei erhält derjenige, der zuerst kommt, eine niedrigere Nummer als diejenigen, die nach ihm kommen. Die Nummern werden anschließend aufsteigend bearbeitet. Dies entspricht exakt dem FIFO-Verfahren, der Struktur einer Queue.

Beide zuvor genannten Fehlvorstellugnen erscheinen besonders passend für die Umsetzung als Item, da sie sich auf häufige und praxisrelevante Themen beziehen. Sie haben zudem gemein, dass sie elementar sind und auch sehr früh im Unterrichtsverlauf auftreten können.

5.5.1. Item 3 - Datenstrukturen

Als Item 3 wird die von Tew u. a. (2005) identifizierte Fehlvorstellung aufgegriffen, die oben bereits beschrieben wurde. Diese besteht daraus, dass Lernende die natürliche Zählweise auf die Datenstruktur Array anwenden und annehmen, dass das erste Element den Index 1 hat. Dementsprechend verschieben sich alle Operationen und das ursprünglich erste Element (d. h. das Element mit dem Index 0) wird ausgelassen.

Aufgabe

Die Lernenden sollen den Wert von *array1* nach der Ausführung des folgenden Programmcodes angeben.

```
1 array1 = [6, 9, 4, 2, 8]
2 array2 = [5, 2, 2, 1]
3
4 array1[2] = array2[3]
5 array1[4] = array2[1]
6 array1[3] = array2[2] + 2
```

Listing 5.3: Code Beispiel Algorithmen 2

Teil A: Welchen Wert hat die Variable array1 nun?
Bitte wählen Sie nur eine der folgenden Antworten aus:

- ☐ [6, 9, 1, 2, 9]
- ☐ [6, 9, 1, 4, 2]
- ☐ [6, 2, 4, 2, 9]
- ☐ [2, 6, 4, 1, 4]

Teil B: Die Antwort eines Lernenden lautet:

array1 = [6, 2, 4, 5, 8]

Welche Fehlvorstellung könnte hier vorliegen?

Musterlösung

Richtige Lösungsschritte:

Schritt 1: array1 = [6, 9, 4, 2, 8]
Schritt 2: array1 = [6, 9, **1**, 2, 8]
Schritt 3: array1 = [6, 9, 1, 2, **2**]
Schritt 4: array1 = [6, 9, 1, **4**, 2]

Die Lösung lautete somit: [6, 9, 1, 4, 2]

Die falsche Antwort des Schülers wird durch die falsche Indexierung des Arrays erzeugt. Das erste Element hat bei ihm die Nummer 1, somit ergeben sich die folgenden falschen Lösungsschritte:

Schritt 1: array1 = [6, 9, 4, 2, 8]
Schritt 2: array1 = [6, **2**, 4, 2, 8]
Schritt 3: array1 = [6, 2, 4, **5**, 8]
Schritt 4: array1 = [6, 2, **4**, 5, 8]

Die Schülerantwort lautet: array1 = [6, 2, 4, 5, 8]

Vorbewertung des Items

Bei diesem Item muss, ähnlich wie zuvor in Item 2, zunächst die richtige fachwissenschaftliche Lösung angegeben werden, bevor die Frage zu den Fehlvorstellungen beantwortet wird. Eine Hürde mag sein, die einzelnen Schritte auf dem Bildschirm nachzuvollziehen und die Operationen „im Kopf“ zu verfolgen. Aufgrund der Länge des Arrays (fünf Elemente und damit kürzer als die Beispielaufgaben in der oben genannten Quelle) bleibt dies aber in einem überschaubaren Umfang. Als eine Art Abkürzung lässt sich die Lösung auch herleiten, indem einzelne Ziffern überprüft werden. So existiert nur eine mögliche Lösung, deren letztes Element eine 2 ist. Ähnliches gilt für den zweiten Teil der Aufgabe. Da in der Schülerantwort das zweite Element verändert wurde, aber keine Zuweisung $array1[1]$ vorhanden ist, lässt sich schnell auf die richtige Lösung schließen. Durch diese Schwierigkeit sollte die richtige Beantwortung somit nicht beeinflusst werden.

Ein Aspekt, der beim Test der Aufgabe genannt wurde, ist die Möglichkeit dass eine Programmiersprache verwandt wird, bei dem die Zählweise des Arrays tatsächlich

mit dem Index 1 beginnt. Dies scheint jedoch zu vernachlässigen zu sein, da alle aktuellen und häufig in der Lehre genutzten Sprachen die Zählweise ab 0 verwenden, darunter C++, Java, JavaScript, PHP und Visual Basic. Gegenbeispiele sind Fortran, COBOL und Smalltalk. Dass einem Testteilnehmer lediglich eine dieser Sprachen bekannt ist, die zuvor genannten allerdings nicht, scheint unwahrscheinlich. Gleichzeitig handelt es sich hier um ein offenes Frageformat, so dass die Probanden die Möglichkeit haben, dies zu kommentieren.

5.5.2. Item 4 - Datentypen, Queue

Aufgabe

Teil A: Wählen Sie die Analogie, aus die am besten eine Warteschlange (queue) repräsentiert und begründen Sie ihre Antwort:

- ☐ Warteschlange an einer Supermarktkasse
- ☐ Vergabe von aufsteigenden Nummern beim Betreten eines Wartezimmers
- ☐ Ausliefern der Bestellungen in einem Restaurant

Teil B Begründen Sie Ihre Entscheidung.

Musterlösung

Die richtige Antwort ist Option 2, die Vergabe von aufsteigenden Nummern an die Wartenden. Dies ist darin begründet, dass dies das eindeutigste FIFO-Prinzip (First In – First Out) ist. Jede Person erhält beim Betreten des Wartezimmers eine eindeutige Nummer, die dann der Reihe nach abgearbeitet werden. Diese Analogie ist insbesondere gut geeignet, da hier der Schwerpunkt auf der tatsächlichen Struktur und ihrer Funktionsweise liegt.

Die erste Antwortoption ist zwar durch die Parallele zum Namen naheliegend, kann aber für Lernende irreführend sein. So bewegen sich die Objekte in einer Warteschlange: nachdem ein Kunde bezahlt und den Kassenraum verlassen hat, rücken die anderen nach. In der Praxis verschieben sich dann alle Objekte in der Schlange. Zusätzlich kann sich die Reihenfolge verschieben, wenn z. B. eine zusätzliche Kasse geöffnet wird.

Die dritte Antwortoption ist zwar der Vergabe von Wartenummern ähnlich (hier steht wieder der Ablauf im Mittelpunkt), allerdings liegt hier nicht zwangsläufig ein FIFO vor. So können z. B. einige Bestellungen früher fertig sein als andere oder bestimmte Bestellungen werden zunächst gesammelt und dann gleichzeitig zubereitet.

Vorbewertung des Items

Dieses Item kann in Hinblick auf das Vorwissen der Befragten irreführend sein. Durch den Namen der Datenstruktur Warteschlange ist naheliegend, dass diese ebenso die passende Analogie ist. Nicht zuletzt ist dies ein oft verwendetes Beispiel in der Fachliteratur und in Schulbüchern. Interessanterweise wird das aber, vermutlich ähnlich häufig, von Lehrenden kritisiert, weil es zwar ein offensichtlicher Vergleich ist, dieser allerdings in mehreren Details nicht stimmig ist. Gleichzeitig muss bedacht werden, dass die Aufgabenstellung sehr subjektiv wirken kann, insbesondere wenn die Befragten eine der genannten Analogien in der Lehre eingesetzt haben und keine Fehlvorstellungen oder Nachteile beobachtet haben.

Ein wichtiger Aspekt dieser Frage ist sicherlich, dass diese ein Multiple-Choice-Format hat. So kann der Testteilnehmer zunächst die Antwortoptionen vergleichen, bevor er eine Antwort gibt. Auf diesem Weg werden vermutlich einige Probanden zu

einer anderen Antwort tendieren, die bei einem offenen Format die Warteschlange als Antwort genannt hätten. Dies ist allerdings nicht zwingend als Nachteil anzusehen, da die Befragten so dazu motiviert werden können, ihre Antwort noch einmal zu reflektieren (vgl. Abschnitt 5.2).

5.6. Logik

Logik ist ein wichtiges Element in vielen Themengebieten der Informatik. Dies wird durch die Bewertung des Horizontalkriteriums der Fundamentalen Ideen durch die Experten unterstützt (vgl. Unterabschnitt 4.2.2). Almstrum (1994) vertritt die These, dass ein grundlegendes Verständnis von Logik die Aneignung von informatischen Fähig- und Fertigkeiten deutlich beschleunigt. In ihrer Studie beschäftigt sie sich insbesondere mit der Frage, ob Lernende im Vergleich mit anderen Konzepten der Informatik besondere Schwierigkeiten mit der Logik haben. Dazu lässt sie Testitems hinsichtlich ihres Bezugs zu Logik bewerten und kommt zu dem Schluß, dass Lernende bei den Aufgaben signifikant schlechter abschneiden, die eine hohe Bewertung erhalten haben.

Cheng u. Holyoak (1985) beschreiben, dass weniger als 10 % der von ihnen getesteten College Studenten in der Lage sind, „if p then q“ und „p if and only if q“ in der praktischen Anwendung zu verstehen. Sie nutzten dazu vier Spielkarten, die mit Buchstaben (A oder B) auf der einen und mit Zahlen (4 oder 7) auf der anderen Seite beschriftet sind. Es wird die folgende Regel genannt: „Wenn eine Karte einen Vokal auf der einen Seite hat, hat sie eine gerade Zahl auf der anderen Seite.“. Die Studierenden wurden daraufhin gefragt, welche Karten umgedreht werden müssen, um diese Regel zu beweisen (in ähnlicher Form bekannt als „Wason's selection task“). Weniger als 10 % der College Studenten gaben hier als richtige Antwort, dass lediglich die Karten mit A und 7 umgedreht werden müssen.

Epp (2003) geht in besonderem Maße auf den Unterschied zwischen Alltagssprache und mathematischer Logik ein. Ein Bikonditional wird dabei häufig als „wenn, dann“-Aussage verfremdet. Sie nennt als Beispiel Eltern, die ihrem Kind vermitteln wollen, dass es nur und nur dann in das Kino gehen darf, wenn es die Hausaufgaben gemacht hat (*if and only if*). Häufig wird dies als „Wenn du deine Hausaufgaben gemacht hast, darfst du in das Kino gehen“ oder „Du darfst in das Kino gehen, wenn du deine Hausaufgaben gemacht hast“ formuliert. Jedoch ist diese Aussagen aus Sicht der Logik eine Implikation, die nicht das ausdrückt, was die Eltern tatsächlich vermitteln wollen. Nach den Erfahrungen von Epp zeigt sich häufig, dass

Lernende davon profitieren, wenn Logik nicht nur in abstrakter Form (d. h. durch Formeln und Wahrheitstabellen) unterrichtet wird, sondern mit der natürlichen Sprache verknüpft wird. So kann vermieden werden, dass falsche Rückschlüsse zur tatsächlichen Bedeutung der Aussagen gezogen werden (Epp, 2003, S. 895).

Herman u. a. (2008) befragte Lernende zu ihrem Verständnis von Aussagenlogik und maß ihre Fähigkeit, natürlichsprachliche Beschreibungen in Boole'sche Ausdrücke, sowie Boole'sche Ausdrücke in natürliche Sprache zu übersetzen. Diese Studie ist Teil eines Projektes, in dem ein CI entwickelt werden soll. Zur Identifikation von Fehlvorstellungen wurden dabei Think-Aloud-Interviews durchgeführt, in denen die Studierenden drei praktische Aufgaben bearbeiten mussten. Es zeigten sich gute Ergebnisse bei der Anwendung und Interpretation einfacher Operatoren (AND, OR und XOR). Auffällig war jedoch die Tendenz, die als schwieriger empfundenen Operatoren auf diese zu reduzieren. Auch hier wird deutlich, dass die Lernenden Probleme haben, logische Operatoren in die natürliche Sprache zu übersetzen. Zu den von Epp (2003) bereits dargestellten Schwierigkeiten kommt hinzu, dass die Konsequenz logischer Ausdrücke falsch dargestellt wird. So wurde if-then häufig zeitlich interpretiert („Zuerst tritt A ein, dann passiert B . . . “).

Vandegrift u. a. (2010) nahmen diese Ergebnisse auf und beschäftigten sich mit dem Verständnis der Logik von Lernenden im Anfangsunterricht. Sie definierten hierzu Regeln für das Backen von Apfelkuchen (z. B. „Do not use both allspice and nutmeg simultaneously“). Daraufhin sollten vorgegebenen Rezepten auf die Einhaltung dieser Regeln überprüft werden. Schwerpunkt dabei ist die Verneinung einer Konjunktion (NAND) und die Äquivalenz (if-and-only-if, auch iff). In einer Online-Umfrage, die die Anwendung in Form einer praktischen Anwendung der Regeln beinhaltete, zeigten lediglich 23 % der Befragten ein vollständiges Verständnis von iff. Die Mehrheit (64 %) interpretierte es als Implikation. Dies ergänzt die Ergebnisse, die Epp (2003) zuvor publiziert hatte, nach denen die Lernenden eine Implikation als Äquivalenz interpretieren.

Die zuvor genannten Beispiele zeigen, dass das Wissen über Logik ein problematisches Gebiet ist. Irritierend aus Sicht des Lehrenden ist hier besonders, wenn Wahrheitstabellen zwar korrekt von den Lernenden ausgefüllt werden können, aber ihre Bedeutung nicht bekannt ist bzw. eine aus fachlicher Sicht falsche Alltagslogik zur Interpretation herangezogen wird. Die Kompetenz, natürlichsprachlich formulierte Regeln, die logischen Verknüpfungen entsprechen, korrekt anzuwenden ist dabei auch für den Lehrenden sehr wichtig. Aus diesem Grund konzentriert sich das folgende Item auf den Vergleich von Regeln für die Zusammenstellung von Backzutaten, die jeweils mit einer Liste von Zutaten abgeglichen werden müssen.

5.6.1. Item 5 - Logik

Aufgabe

Für die Zubereitung eines Auflaufs gelten die folgenden Regeln:

1. Wenn der Auflauf Nudeln enthält, muss er Käse enthalten.
2. Es ist nicht möglich, dass der Auflauf von den Zutaten Brokkoli und Schinken entweder nur Brokkoli oder nur Schinken enthält.
3. Der Auflauf enthält nicht gleichzeitig Kartoffeln und Hackfleisch.

Entsprechen die folgenden Zutatenlisten den Regeln? Falls nicht, gegen welche Regel verstoßen sie?
Teil A: Zutatenliste 1: Nudeln, Schinken, Brokkoli

- ☐ Richtig
- ☐ Regel 1
- ☐ Regel 2
- ☐ Regel 3

Teil B: Zutatenliste 2: Nudeln, Hackfleisch, Brokkoli, Käse

- ☐ Richtig
- ☐ Regel 1
- ☐ Regel 2
- ☐ Regel 3

Teil C: Zutatenliste 3: Kartoffeln, Brokkoli, Schinken, Käse

- ☐ Richtig
- ☐ Regel 1
- ☐ Regel 2
- ☐ Regel 3

Musterlösung

1. Regel 1
2. Regel 2
3. Richtig

Vorbewertung des Items

Diese Aufgabe stellt, nach den Erfahrungen der Pretests (vgl. Unterabschnitt 5.2.1) zu urteilen, eine der größten Herausforderungen des Testinstruments in fachlicher Hinsicht dar und es ist zu erwarten, dass hier niedrige Punktzahlen erzielt werden. Dies ist zum einen im Themengebiet Logik begründet, das, wie zuvor bereits mehrfach dargestellt, grundsätzlich als schwierig bewertet werden kann. Zum anderen ist die Logik zwar Teil des Informatikunterrichts in der Schule und der Einführungsveranstaltungen in Universitäten, allerdings selten aus der hier eingenommenen Perspektive. Zumeist werden Wahrheitstabellen aufgestellt, die strikt nur die Aussagen betrachten und diesen ein Resultat zuordnen. Die Beschreibung in Alltagssprache bzw. überhaupt die Versprachlichung, erfolgt kaum. Insofern zeigen sich hier zwei Herausforderungen, die zu meistern sind.

Zudem ist dieses Item als rekursive Aufgabe konstruiert. Das heißt, dass eine Lösung einer fachwissenschaftlichen Aufgabe präsentiert wird, welche analysiert werden soll. Ausgehend davon ergibt sich als Anforderung zur erfolgreichen Bearbeitung des Items, dass die Aufgabenstellung verstanden und die Fragestellung selbstständig bearbeitet werden muss. Dies gibt dem Item, insbesondere im Vergleich zu Items, die keine konkrete Aufgabenstellung einschließen, einen höheren fachlichen Anspruch.

5.7. Programmierung - Software

Die Programmierung ist ein sehr umfangreiches Gebiet und lässt sich nur schwerlich eingrenzen. Sie ist heute als das grundlegende Thema (in Form einer Vorlesung bzw. Unterrichtseinheit) zu Beginn der Informatikausbildung anzusehen und wird nicht nur dadurch oft von außen als Essenz der Informatik verstanden.

Da die Konzepte, die in der Erhebung in Kapitel 4 gemessen wurden, nicht trennscharf gewählt wurden, lassen sich diverse ebenso gemessene Unterthemen dem Konzept der Programmierung zuordnen. Ein Blick in das ACM Curriculum zeigt insbesondere aus Anwendungsperspektive eine starke Überschneidung mit dem Bereich Software. Dies bestätigen die Autoren:

> „For example, even a fairly straightforward introductory course sequence will likely augment material from SDF [„Software Development Fundamentals“, Anmerkung der Autorin] with topics from the Programming Languages Knowledge Area related to the choice of language used in

> the course and potentially some concepts from Software Engineering.“ (ACM/IEEE-CS Joint Task Force on Computing Curricula, 2013, S. 45)

Im Gebiet der „Programming Languages“ werden in den grundlegenden Kursen konkrete Programmierparadigmen (objektorientiert und funktional, insgesamt sieben Einheiten) und Typisierung (eine Einheit) behandelt. Übergreifend betrachtet sind hier die Werkzeuge im Mittelpunkt, die für die Softwareentwicklung nötig sind. Themen wie Syntax und Semantik, Datenstrukturen oder Rekursion lassen sich nur mit Hilfe einer Programmiersprache sinnvoll lehren. Aus dieser Perspektive fällt eine Abgrenzung der Fehlvorstellungen in den beiden Gebieten schwer und die zuvor identifizierten Konzepte *Programmierung* und *Software* werden im Folgenden zusammen betrachtet.

Sirkiä u. Sorva (2012) untersuchten Fehlvorstellungen mit Hilfe einer Visual Program Simulation (VPS). Dabei handelt es sich um ein graphisches Interface, mit dem interaktiv der Ablauf eines Programms durchlaufen wird. Ein besonderer Vorteil dabei ist, dass hier automatisiert Feedback zu bekannten Fehlern gegeben werden kann. Des Weiteren werden die einzelnen Schritte durch Animationen veranschaulicht. Die Studie basiert auf einem eigens entwickelten System namens *UUhistle*. Sie hat das Ziel, dieses um zusätzliches Feedback zu häufigen Fehlern und Fehlvorstellungen zu erweitern. Zu diesem Zweck wurden 24.000 Lösungen von Lernenden gesammelt und analysiert. Aufgrund der Zielgruppe der VPS handelt es sich dabei insbesondere um grundlegende, konzeptuelle Fehlvorstellungen, die durch die Vielzahl der dokumentierten Kontexte sehr gut dokumentiert sind. Sehr vielversprechend ist, dass mehr als 200 Fehler von jeweils mehr als 10 Schülern gemacht wurden. Dies verdeutlicht, dass es möglich ist, trotz der automatisierten Erhebung Fehlermuster zu erkennen. Dementsprechend konnten sie auch diversen Fehlvorstellungen zugeordnet werden.

Eine häufig im Zusammenhang mit grundlegenden Kontrollstrukturen genannte Fehlvorstellung bei Anfängern ist, dass sie davon ausgehen, dass eine if-else-Verzweigung stets komplett durchlaufen wird (z. B. Sorva, 2012). Sie beobachten, dass der Programmcode Zeile für Zeile durchlaufen wird und schließen dadurch offensichtlich aus, dass es darin Sprünge oder Verzweigungen gibt. Dies steht in Zusammenhang mit dem als „superbug“ bezeichneten Phänomen, dass die Tendenz von Lernenden beschreibt, dem Computer eine menschliche Intelligenz zuzusprechen, die es ihm ermöglicht, Entscheidungen zu treffen, die nicht im Programmcode definiert sind. Gleichzeitig fällt es Lernenden zu Beginn der Informatikausbildung oftmals schwer, Probleme so zu definieren, dass alle benötigten Informationen

gegeben sind. Während dies bei einer Kommunikation mit einem Menschen zu einer Rückfrage führt, ist das Ergebnis bei der Programmierung eine Fehlermeldung, die für Anfänger oft nicht leicht zu interpretieren ist (Pea, 1986; Saeli u. a., 2011).

Auch verschwimmt zu Beginn der Informatikausbildung oftmals das Wissen über mathematische Variablen und die Art und Weise, wie Variablen in der Informatik genutzt werden (z. B. Doukakis u. a., 2007; Oldenburg, 2012). Es fällt den Lernenden beispielsweise schwer, die Bedeutung des Gleichheitszeichens richtig zu verstehen. So wird die typische Schreibweise arithmetischer Formeln übernommen und die Rechenoperation auf die linke Seite einer Zuweisung geschrieben (1 + 2 = Summe). Eine ähnliche Fehlvorstellung steht hinter der Interpretation, dass eine Zuweisung wie $x = x+1$ nicht möglich ist, da die Gleichung mathematisch nicht lösbar ist. Eine weitere interessante Fehlvorstellung ist der Gedanke, dass eine Variable in einem Programmcode, ebenso wie in einer mathematische Gleichung, mehrere richtige Werte gleichzeitig haben kann (z. B. $x^2 = 4$ mit $x = 2$ und $x = -2$).

Das folgende Item greift zwei der zuvor genannten Fehlvorstellungen auf. Zum einen die im vorherigen Absatz erwähnte Schreibweise bei der Deklaration von Variablen in Teil A. Zum anderen die Vorstellung, dass eine if-else-Verzweigung vollständig durchlaufen und trotzdem korrekt interpretiert wird in Teil B.

5.7.1. Item 6 - Programmierung, Code

Mathematische Behandlung von Variablen

Aufgabe Teil A

Das folgende fehlerhafte Programm, welches die Summe zweier eingegebener Zahlen ermitteln soll, wurde von einem Schüler geschrieben.

```
1 x = input("Enter number 1: ")
2 y = input("Enter number 2: ")
3
4 x + y = sum
5
6 print(sum)
```

Welche Fehlvorstellung könnte hier vorhanden sein?

Achtung: Es geht in dieser Aufgabe nicht darum, mögliche Fehler (z.B. Strings als Eingabe) abzufangen.

Musterlösung

Die Zuweisung $x + y = sum$ ist in der falschen Reihenfolge geschrieben worden, wodurch der Variable *sum* kein Wert zugewiesen und eine Fehlermeldung ausgegeben wird. Diese Fehlvorstellung entsteht, weil Lernende ihre Denkweise aus der Mathematik in die Programmierung übertragen. Hier werden im Regelfall die zu addierenden Zahlen auf die linke Seite des Gleichheitszeichens geschrieben und das Ergebnis auf die rechte Seite. Sie haben somit ein mentales Modell dazu gebildet, welche Funktion ein Gleichheitszeichen hat und weichen bei der Aufgabenlösung nicht davon ab.

if und else Statements werden ausgeführt

Aufgabe Teil B

Ihr Schüler hat ein Programm erstellt, welches ausgeben soll, ob eine Zahl gerade oder ungerade ist. Er versteht nicht, warum er nur bei ungerade Zahlen eine Ausgabe sieht.

```
1 int zahl = 6;
2
3 if (zahl%2 == 0) {
4   x = "gerade" }
5 else {
6   y = "ungerade"
7   print(x)
8   print(y)
9  }
```

Welche Fehlvorstellung könnte hier vorhanden sein?

Musterlösung Teil B

Der Schüler nimmt an, dass beide Blöcke in der if-else-Anweisung ausgeführt werden. Diese Vorstellung entsteht, weil die Lernenden oft zu Beginn des Unterrichts vermittelt bekommen, dass stets der komplette Code, Zeile für Zeile, durchlaufen wird. Dass hier allerdings durch die Kontrollstruktur ein Teil übersprungen wird, wird nicht erkannt.

Ein weiterer Aspekt ist, dass dem Compiler bzw. Interpreter eine Intelligenz zugesprochen wird, so dass dieser auf Basis des Codes „wissen“ kann, wie vorzugehen ist. In diesem Fall hätte er aus der if-else-Anweisung interpretiert, ob die Zahl gerade oder ungerade ist und würde dementsprechend das richtige Ergebnis ausgeben.

Vorbewertung des Items

Beide Aufgabenteile behandeln Fehlvorstellungen, die klar aus dem Programmcode ersichtlich sind. Im ersten Teil wird die Variable nicht in der üblichen, bekannten Form deklariert. Im zweiten Teil befinden sich beide Ausgabebefehle innerhalb eines

Teils des if-else-Statements und können damit nicht in jedem der beiden möglichen Fälle ausgegeben werden. Dies sind Anfängerfehler, die bei einem Grundverständnis für Programmcode nicht auftreten sollten bzw. sofort erkannt und korrigiert werden könnten. Aus diesem Grund wurde bei diesem Item darauf verzichtet, vorab explizit die fachwissenschaftlich korrekte Lösung abzufragen.

Die Interpretation, warum diese Fehler entstanden sind, ist allerdings schwieriger. Das Muster in Aufgabenteil A dürfte dabei allerdings leicht erkennen zu sein. Bei ersten Tests, die während der Entwicklung durchgeführt wurden, wurde dieses Item jeweils als einfach eingestuft. Der Fehler als solches wurde schnell erkannt und konnte schnell interpretiert werden.

Teil B erscheint herausfordernder, da hier auf den ersten Blick ein simpler Fehler vorzuliegen scheint, durch den der erste Ausgabebefehl außerhalb der ersten Klammer steht. Da er aber zusammen mit dem zweiten Ausgabebefehl steht und nicht direkt hinter der geschlossenen Klammer, ist dies unwahrscheinlich. Eine nachvollziehbare Möglichkeit, wie das richtige Ergebnis ausgegeben werden kann, ist somit das Durchlaufen des else-Teils, auch wenn die if-Bedingung erfüllt ist.

5.8. Systems

Um den Bereich *Systems* zu überblicken, soll dieser zunächst definiert und eingeordnet werden. Das ACM Curriclum formuliert die Bedeutung wie folgt:

> „Computer systems broadly span the subdisciplines of operating systems, parallel and distributed systems, communications networks, and computer architecture.“ (ACM/IEEE-CS Joint Task Force on Computing Curricula, 2013, S. 186).

„Systems“ wird hier besonders in zwei Bereichen erwähnt. Dies sind „Operating Systems“ und „Systems Fundamentals“.

Hammond u. Rogers (2006) führten mit 11-jährigen Schülern Interviews durch, in denen diese nach ihrem grundlegenden Verständnis von Computersystemen, konkreten Anwendungsszenarien (u. a. in ein System einloggen und Dateien speichern) und Funktionsweisen befragt wurden. Dabei zeigte sich, dass alle Schüler ein Verständnis für einfache Funktionsweisen („input/output“, „storage“) hatten und grundlegende Kompetenzen bei der Benutzung des Computers. Ihr Wissen ist allerdings durch ihre eigenen Erfahrungen bzw. Interpretation von Zusammenhängen begrenzt. So

bestand z. B. die Vorstellung, dass sich eine Maus grundsätzlich auf der rechten Seite des Computers befinden muss.

Pamplona u. a. (2013) führten eine Studie in einem Kurs einer Online Universität durch, um einerseits Konzepte zu Betriebssystemen zu ermitteln, die am schwierigsten zu verstehen sind und andererseits alternative Konzepte (im Sinne von Fehlvorstellungen) zu diesen zu finden. Die Lehre in diesem Kurs basierte lediglich auf Kursunterlagen und Aufgaben, sowie die Möglichkeit, den Professor bei Fragen telefonisch zu kontaktieren. Präsenzveranstaltungen oder Vorlesungsvideos gab es nicht. Für die Befragung wurden drei Fragebögen zu den Themen des Kurses mit jeweils zehn Multiple-Choice-Fragen, bei denen die gewählte Antwort jeweils begründet werden musste, entworfen. Die Antworten wurden anschließend ausgewertet und codiert. Als Ergebnis zeigten sich insbesondere Fehlvorstellungen im Bereich der Prozessverwaltung von Prozessoren und paralleler Programmierung. Anzumerken ist zu dieser Studie allerdings, dass sie mit lediglich neun Probanden durchgeführt wurde, die eine sehr eingeschränkte bzw. spezielle Form der Lehre durchlaufen hatten. Hier stellt sich daher die Frage der Validität und Relevanz der Ergebnisse.

Ioannou u. Angeli (2014) untersuchten Fehlvorstellungen über den Hauptprozessor (CPU), indem sie Materialien und Aufgaben zu diesem Thema entwickelten. Als beispielhafte Fehlvorstellungen werden hier z. B. die Gleichsetzung der CPU mit dem gesamten Computer und die Vorstellung, dass die CPU sämtliche Aktionen des Computers beobachtet und verarbeitet, z. B. Tastatureingaben oder Druckvorgänge. Das Thema entstammt einer früheren Studie (vgl. Tabelle 4.1, Ioannou u. Angeli (2013)). Die Testfragen wurden hier im Multiple-Choice-Format formuliert.

Die einzelnen Komponenten eines Computers scheinen studienübergreifend mit Fehlvorstellungen verbunden zu sein. Dies ist aufgrund des selten hardwareorientierten Ansatzes in Einführungsveranstaltungen nicht verwunderlich. Die Lernenden kommen bei der Programmierung selten in Berührung mit Hardwarekomponenten, da auch Assemblersprachen heute eher selten unterrichtet werden. Häufig sind den Lernenden nur die Funktionen, bzw. simple Analogien bekannt. Auf letztere soll sich das folgende Item beziehen.

5.8.1. Item 7 - Systeme, Analogien

Aufgabe 1

Analogien werden in der Informatik häufig verwendet, um unbekannte Konzepte zu erklären und den Lernprozess zu fördern.

Der Arbeitsspeicher eines Computers kann als Eis (im Sinne von gefrorenem Wasser) beschrieben werden. Es hat, ebenso wie Speicher, zwei Zustände: flüssig und gefroren. Um zwischen diesen zu wechseln, braucht man Energie. Wenn Eis schmilzt, geht der vorherige Zustand (d.h. die Speicherbelegung) verloren.

Nennen Sie mindestens eine Eigenschaft von Arbeitsspeicher, für die sich diese Analogie nicht eignet und daher zu einer Fehlvorstellung bei Lernenden führen kann. Begründen Sie ihre Antwort kurz.

Musterlösung

Mögliche Ideen sind:

- Der Speicherinhalt wird komplett gelöscht, sobald die Stromversorgung getrennt wird. Es ist sogenannter „flüchtiger Speicher“. Eis verändert lediglich seinen Aggregatzustand.
- Eis wirkt unstrukturiert und unsortiert. Es ist schwierig sich vorzustellen, dass in diesem Konstrukt Speicheradressen existieren, auf die exakt zugegriffen werden kann.

Vorbewertung des Items

Dieses Item ist eindeutig dass offenste und erfordert die meiste Erfahrung und Kreativität bei der Beantwortung. Die Bewertung stellt daher die größte Herausforderung aller Items dar (vgl. Abschnitt 5.2). Der Grund hierfür ist die Tatsache, dass es vermutlich keine eindeutig bewertbaren richtigen bzw. falschen Antworten gibt. Vielmehr sollte es hier darum gehen, sich in die Position des Schülers zu versetzen und daraus mögliche Fehlvorstellungen abzuleiten. Es ist daher zu vermuten, dass die überwiegende Mehrheit der Antworten als richtig bewertet werden kann. Dies bedeutet allerdings nicht, dass Fehlvorstellungen hier besonders leicht zu erkennen

sind, sondern dass diese durch das transferierte Alltagswissen sehr vielfältig sei können (vgl. Kapitel 2). In den Vortests hat sich allerdings gezeigt, dass dieses Item häufig gar nicht beantwortet werden konnte. In diesem Fall ist die Bewertung somit ebenso eindeutig.

6. Hauptstudie

6.1. Überblick

Die zuvor in Kapitel 5 vorgestellten Testitems wurden nach mehreren Testdurchläufen mit ausgewählten Personen in einer breiten Studie eingesetzt. In diesem Kapitel soll nun zunächst ein Überblick über die dabei geltenden Rahmenbedingungen und die Durchführung gegeben werden. Daraufhin folgt die deskriptive statistische Analyse und Interpretation hinsichtlich verschiedener Kriterien und Aspekte, die mit beispielhaften Antworten der Probanden ergänzt wird. Abgeschlossen wird dieses Kapitel mit einer Zusammenfassung, in der die Ergebnisse noch einmal in ihrer Gesamtheit betrachtet werden.

6.2. Rahmenbedingungen

Die Umfrage wurde, ähnlich wie die zuvor durchgeführte Erhebung zu informatischen Konzepten, mit Hilfe der Umfrage-Software LimeSurvey implementiert. Dabei wurden zu Anfang einige demographische Fragen gestellt, deren Beantwortung optional war (vgl. Unterabschnitt 6.2.1 für weitere Details). Diese wurden allerdings von allen Testteilnehmern vollständig beantwortet.

Darauf folgten die Testitems in randomisierter Reihenfolge. Diese Anordnung liegt darin begründet, dass die Items unterschiedliche Bereiche abfragen und dabei sowohl fachwissenschaftliche, als auch fachdidaktische Schwerpunkte setzen. Da die Gruppen der befragten Personen hinsichtlich ihrer beruflichen Interessen breit gestreut ist, soll die Variierung der Reihenfolge sicherstellen, dass keine der befragten Gruppen frühzeitig abgeschreckt wird bzw. die Umfrage systematisch von einer Gruppe bei einem bestimmten Testitem beendet wird und die nachfolgenden Items nicht beantwortet werden. Diese Entscheidung wurde getroffen, nachdem bei den

Testdurchläufen mehrmals entsprechende Kommentare geäussert wurden. Diese entstanden insbesondere beim Umgang von Teilnehmern ohne fachdidaktische Ausbildung mit fachdidaktischen Fragen (im Sinne von „Ich bin kein Lehrer, das kann ich nicht wissen!“).

6.2.1. Demographische Daten

Zu Beginn wurden, wie bereits zuvor erwähnt, einige Fragen zu demographischen Daten gestellt. Hier wurde zunächst das Alter und die Zugehörigkeit zu einer der drei eingeladenen Gruppen abgefragt. Es folgten eine Frage zur abgeschlossenen Ausbildung, zur Lehrerfahrung und zu einer eventuell vorhandenen didaktischen Ausbildung. Ausserdem sollten die Teilnehmer angeben, ob sie Erfahrung im Forschungsgebiet der Didaktik der Informatik haben.

6.3. Durchführung

Die Einladungen zur Umfrage wurden ab dem 19.11.2015 via E-Mail verschickt. Des Weiteren wurden mehrere Multiplikatoren gebeten, die Einladung weiter an geeignete Personen zu verteilen. Aus diesem Grund lässt sich nicht nachvollziehen, wie viel Prozent der Eingeladenen tatsächlich teilgenommen haben. Aufgrund der zugesicherten Anonymität wurden in Limesurvey keine personalisierten Teilnahmeschlüssel generiert, die dies ermöglichen würden.

Es wurde ausserdem explizit darauf Wert gelegt, dass nur Personen, die bislang nicht mit dem Forschungskontext bzw. den konkreten Forschungsansätzen dieser Arbeit vertraut sind oder sogar in diesen involviert waren, eingeladen werden. Nur so kann sichergestellt werden, dass einzelne Fehlvorstellungen nicht bereits auf diesem Weg kennengelernt wurden und aus diesem Grund bei der Beantwortung der Items richtig beschrieben oder identifiziert wurden. Dies hat zur Folge, dass mehrere in der Fachdidaktik Informatik tätige Lehrer, die vorab großes Interesse an der Studie geäussert hatten, nicht eingeladen wurden. Dies ist bedauerlich, da aus dieser Gruppe sicherlich zahlreiche interessante Antworten zu erwarten gewesen wären, stellt aber die Objektivität und Repräsentativität der Daten sicher.

6.4. Evaluation

Im Folgenden soll nun die Auswertung des Tests erfolgen. Dies geschieht, indem zunächst die Messgruppe und ihre demographischen Daten vorgestellt werden. Daraufhin werden die Gesamtergebnisse nach Gruppen aufgeteilt präsentiert, bevor die einzelnen Items im Details betrachtet werden.

Die Auswertung verfolgt insbesondere die Beantwortung der Forschungsfrage, inwieweit das Wissen in den einzelnen Gruppen ausgeprägt ist bzw. ob zwischen den Gruppen signifikante Unterschiede bestehen. Gleichermaßen soll eine Verifizierung der Items, sowohl auf rein statistischer Basis, als auch durch eine subjektive Betrachtung von Antworten, erfolgen.

6.4.1. Teilnehmer

An der Erhebung nahmen insgesamt 73 Probanden teil, darunter 24 Studierende im Fach Lehramt Informatik, 26 Informatik Lehrer und 23 Teilnehmer, die in sonstigen Berufen im Gebiet der Informatik beschäftigt sind.

Von den befragten Lehrern haben 24 ein abgeschlossenes Studium Lehramt Informatik, jeweils eine Person hat eine didaktische Weiterbildung durchlaufen und eine hat keine didaktische Ausbildung. Die Altersstruktur entspricht den Erwartungen: während die Mehrheit der Studierenden zwischen 20 und 29 Jahre alt ist (21 Teilnehmer), liegt die Mehrheit der Lehrer und Sonstigen bei 30 bis 39 Jahren. Lediglich ein Teilnehmer (ein Lehrer) ist zwischen 50 und 59 Jahre alt. Hinsichtlich der Lehrerfahrung ist erwähnenswert, dass immerhin sechs Studierende bereits Informatikunterricht durchgeführt haben, davon sogar ein Teilnehmer länger als ein Jahr. Keiner der Kandidaten gab an, im Forschungsgebiet Fachdidaktik Informatik aktiv gewesen zu sein.

Die Anzahl der Probanden in den einzelnen Gruppen ist ausreichend, um die Anforderungen zur Erzielung signifikanter Ergebnisse in dieser Studie zu erfüllen. Um die minimale Stichprobengröße für einen nichtparametrischen Test zu ermitteln, wird empfohlen, eine Power-Analyse für den äquivalenten parametischen Test durchzuführen und das Ergebnis um 15 % zu erhöhen. Der Power Test für einen einfaktoriellen ANOVA ergibt bei einer Effektstärke von $f = 0.4$, einer Signifikanz von $p = 0.05$ und einer Power von 0.75 für drei Ausprägungen eine minimale Anzahl an Testteilnehmern von $n = 18.93$ für jede Gruppe. Die Anzahl von mindestens 21.77 Probanden in jeder Gruppe ist hier somit erfüllt.

	Gesamt	Studierende	Lehrer	Sonstige
Minimum richtig gelöste Items	0	0	1	5
Maximum richtig gelöste Items	13	13	12	12
$\bar{x}$ (arithm. Mittel)	6.92	4.50	8.00	8.22
$\tilde{x}$ (Median)	7.00	4.00	7.50	8.00
SD	3.24	3.47	2.85	1.73

Tabelle 6.1.: Übersicht der richtig beantworteten Items nach Gruppen

6.4.2. Auswertung Gesamt

An dieser Stelle sollen die Gesamtergebnisse des Tests betrachtet werden. Diese lassen sich auf unterschiedliche Weisen darstellen. Die Konzeption, wie auch die Zusammenstellung der einzelnen Aufgaben legt eine Zusammenfassung bzw. Eingrenzung der Items nahe, um eine Gewichtung einzelner Bereiche zu vermeiden, die die Ergebnisse verfälschen könnte.

Insgesamt bestand der Test aus 13 einzeln zu bewertenden Items. Die maximale Anzahl der gelösten Items in der Erhebung war 13, die minimale Anzahl lag bei keinem Item. Beide Ergebnisse wurden durch einen Studierenden erzielt. Dies ist insofern überraschend, da das durchschnittliche Ergebnis dieser Gruppe deutlich unter dem Wert der beiden anderen Gruppen liegt (4.50 für Studierende zu 8.00 für Lehrer bzw. 8.22 für Sonstige). Die durchschnittliche Anzahl der richtig bearbeiteten Items lag insgesamt bei 6.92 Items. Hier existiert ein statistisch sehr signifikanter Unterschied zwischen den einzelnen Gruppen ($H = 18.777$, $p < .001$). Eine genauere Aufschlüsselung ist in Tabelle 6.1 zu finden.

Diese Auswertung zur Bewertung des Wissens über Fehlvorstellungen und damit zur Beantwortung der Forschungsfragen dieser Arbeit zu verwenden, wäre nicht zielführend, da zum einen der Bereich Logik durch drei einzelne Items stark überrepräsentiert ist und zum anderen die fachwissenschaftlichen Items, die zur Überprüfung der jeweils darauf folgenden Items konzipiert wurden (vgl. Kapitel 5), einbezogen werden würden. Aus diesem Grund wurde das Instrument für die nun folgende Auswertung so angepasst, dass die drei Items zum Thema Logik zu einem Item (d. h. einem Punkt in der Auswertung) zusammengefasst wurden und die drei fachwissenschaftlichen Fragen entfernt wurden. Somit liegt die zu erreichende maximale Punktzahl bei 8 Punkten.

Hier zeigt sich, dass die Lehrer das höchste Ergebnis erzielten ($\bar{x} = 4.92$, $\bar{x} = 2.65$), gefolgt von der Gruppe der Sonstigen ($\bar{x} = 4.35$, $SD = 1.73$) und den Studierenden

	Gesamt	Studierende	Lehrer	Sonstige
Minimum Punkte	0	0	0	2
Maximum Punkte	8	8	8	7
$\bar{x}$ (arithm. Mittel)	3.95	2.54	4.92	4.35
$\tilde{x}$ (Median)	4.00	2.50	5.00	4.00
SD	2.04	3.19	2.65	1.73

Tabelle 6.2.: Auswertung Gesamtpunkte nach Gruppen

($\bar{x} = 2.54$, $SD = 3.19$) Die Unterschiede zwischen den einzelnen Gruppen sind statistisch hoch signifikant ($H = 18.644$, $p < .001$).

Die Reliabilität (Cronbachs α) des Tests beträgt bei acht Items 0.65, was einen zufriedenstellenden Wert für den Umfang der Erhebung darstellt. Der Test ist damit für den Vergleich der gemessenen Gruppen geeignet.

6.4.3. Individuelle Auswertung der Items

Die einzelnen Items wurden zusätzlich individuell ausgewertet. Dazu wurde jeweils ein Chi-Quadrat-Unabhängigkeitstest durchgeführt, um den Zusammenhang zwischen den Ergebnissen der Studierenden, der Lehrer und der Gruppe der Sonstigen zu prüfen. Die Ergebnisse, sowie die Anteile der richtigen Antworten sind Tabelle 6.3 zu entnehmen.

Im Folgenden sollen nun die Ergebnisse grafisch veranschaulicht und interpretiert werden. Die dabei verwendeten Balkendiagramme wurden jeweils so konzipiert, dass die richtige Antwort als erste Antwort in der Legende aufgeführt ist und sich im untersten Bereich der Balken befindet. Dies soll dabei helfen, die Diagramme schneller interpretieren zu können. Zusätzlich werden einzelne Beispiele für Antworten der Probanden gegeben, um ein beschriebenes Antwortverhalten vorzustellen und zu erläutern.

Item 1 - Algorithmen

Das Item zum Thema Algorithmen ist eines von drei Items, bei denen zunächst mit Hilfe einer fachwissenschaftlichen Frage (in Teil A) festgestellt wird, ob der Testteilnehmer die der thematisierten Fehlvorstellung zugrundeliegende Frage selbst

	χ^2 (2)	p	Richtige Antworten (%) nach Gruppen Studierende	Lehrer	Sonstige
AL1	7.98	.018	54.17	69.23	91.30
AL2	0.85	.655	54.17	65.38	65.22
AR1	6.63	.036	62.50	84.62	91.30
AR2	9.09	.011	37.50	76.92	69.57
DS1	14.04	$< .001$	12.50	65.38	47.83
DS2	15.42	$< .001$	8.33	61.54	34.78
L1	8.53	.014	33.33	42.31	73.91
L2	10.17	.006	25.00	26.92	65.22
L3	17.13	$< .001$	20.83	34.62	78.26
PR1	6.25	.044	58.33	88.46	78.26
PR2	10.09	.006	20.83	65.38	47.83
Sort	3.68	.185	41.67	65.38	65.22
Sys	11.12	.004	20.83	53.85	13.04

Tabelle 6.3.: Individuelle Auswertung der Items

beantworten kann, bevor er die falsche Antwort eines Lernenden in Teil B interpretiert. Dies hat sich in diesem Fall als sehr sinnvoll erwiesen, da in mehreren Antworten nicht konkret Bezug auf die Antwort in der Fragestellung genommen wurde bzw. aus der Argumentation nicht ersichtlich war, ob sich diese auf den ersten oder den zweiten vorgestellten Algorithmus bezieht.

Das Ergebnis, dass lediglich 13 von 24 Studierenden die fachwissenschaftliche Frage korrekt beantworten konnten, ist überraschend, da eine simple Effizienzanalyse im Regelfall in Einführungsveranstaltungen in der Informatik behandelt worden sein sollte. Aus den Antworten geht mehrfach hervor, dass hier exakt die Fehlvorstellungen vorhanden sind, die in der Antwort identifiziert werden sollten. Ein Beispiel ist diese Antwort eines Studierenden: *„Algorithmus 2 hat eine zusätzliche Schleife. Das wirkt zunächst wie eine zusätzliche Fallunterscheidung. Was danach passiert ist allerdings identisch. Somit wird also zusätzliche die Schleife ausgeführt.“*.

Bemerkenswert ist weiterhin, dass dies gemeinsam mit dem fachwissenschaftlichen Item zu Arrays das Item mit der höchsten Quote korrekter Antworten einer Gruppe ist. Es wurde von 91.30 % der Gruppe der Sonstigen richtig beantwortet.

Hinsichtlich der Identifikation der Fehlvorstellung zeigen sich bei der Gruppe der Lehrer und der Sonstigen sehr ähnliche Ergebnisse. Erstere erzielten mit 65.38 % richtigen Antworten ein marginal besseres Ergebnis. Die geringe Quote

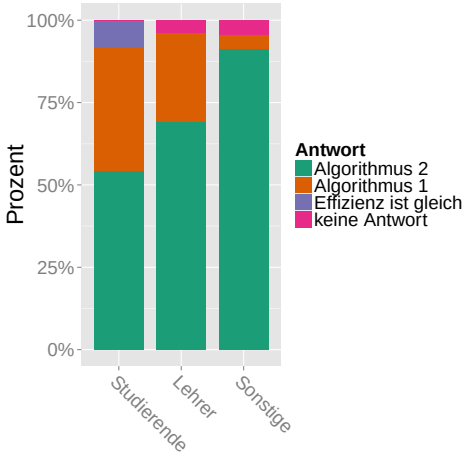


Abbildung 6.1.: Item Algorithmen A - AL1

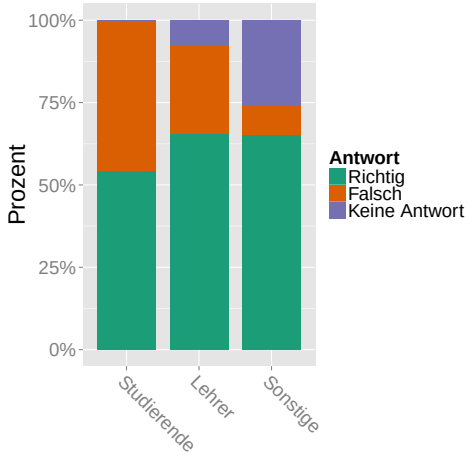


Abbildung 6.2.: Item Algorithmen B - AL2

der richtigen Antworten bei den Studierenden ist in besonderem Maße auf die falsch beantwortete fachwissenschaftliche Frage zurückzuführen. Dies bestätigt sich darin, dass alle Testteilnehmer aus der Gruppe der Studierenden die Frage nach Fehlvorstellungen richtig beantworten konnten, wenn sie zuvor die richtige Antwort auf die fachwissenschaftliche Frage gegeben hatten.

Item 2 - Algorithmen, Sortieren, Datentypen

Dieses Item wurde als offenes Item konzipiert, bei dem die möglichen Antworten eine große thematische Breite haben. Die Mehrzahl der Antworten, die hier als falsch bewertet wurden, bezogen sich auf die Eignung des Sortieralgorithmus bzw. möglicherweise besser geeignete Algorithmen. Ein Beispiel hierfür ist die folgende Antwort eines Studierenden: *„Bubblesort wäre z.B. effektiver“*. Dies ist zweifellos richtig, aber als Antwort auf die Frage nach Fehlvorstellungen nicht angemessen.

Während in der Vorbewertung des Items (vgl. Kapitel 5) davon ausgegangen wurde, dass die Behandlung der Zahlen bei der gegebenen Beschreibung besonders heraussticht, wurde im Test mit einer ähnlichen Häufigkeit die Konzeption und der Aufbau des Algorithmus genannt: *„Hier wird nicht genau beschrieben, wie das Ordnen in die drei Gruppen und wie das Ordnen in den Gruppen ablaufen soll. Meiner Meinung nach liegt hier eine Fehlvorstellung vor, was genau ein*

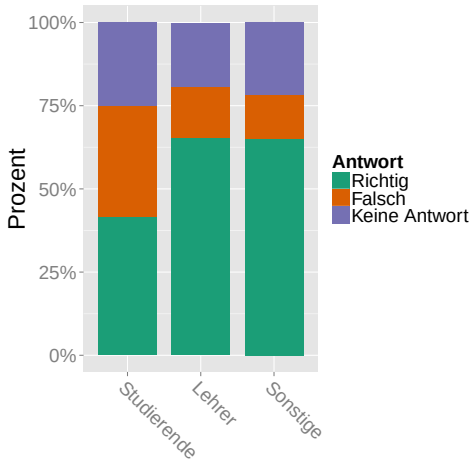


Abbildung 6.3.: Item Sortieralgorithmen (Sort)

Algorithmus ist. Dem Student scheint nicht klar zu sein, dass ein Algorithmus das genaue Vorgehen, Schritt für Schritt, beschreibt.“.

Die Quote der richtigen Antworten entsprach sowohl bei der Gruppe der Lehrer als auch bei der Gruppe der Sonstigen der für das zuvor vorgestellte Item zu Algorithmen. Die individuelle Bepunktung ist allerdings nicht identisch, d. h. nicht jeder Teilnehmer, der eines der Items korrekt beantwortet hat, hat das andere richtig beantwortet.

Wie bereits durch die Punkteverteilung offensichtlich ist, unterscheiden sich die Ergebnisse der einzelnen Gruppen bei diesem Item nicht signifikant (χ^2 (2) = 3.68 und p = .185).

Item 3 - Datenstrukturen, Arrays

Aus den Balkendiagrammen in Abbildung 6.4 wird deutlich, dass dieses Item das Gesamtergebnis des Testinstruments sehr gut widerspiegelt. Während die fachwissenschaftliche Frage im ersten Teil des Items noch von 91,30 % der Teilnehmer aus der Gruppe der Sonstigen richtig gelöst wurde, fallen diese im zweiten Teil hinter die Gruppe der Lehrer zurück.

Das zeigt, dass erstere zwar selbständig die Aufgabe sehr gut lösen können, aber in der Interpretation einer falschen Antwort schlechter abschneiden. Interessant ist

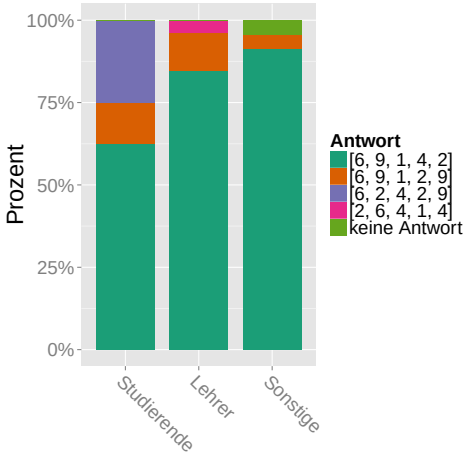


Abbildung 6.4.: Item Arrays A - AR1

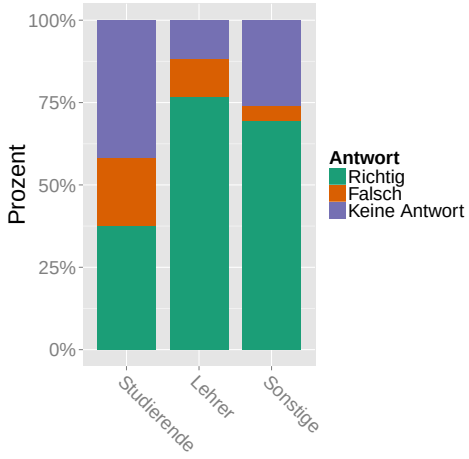


Abbildung 6.5.: Item Arrays B - AR2

hier, dass mehrere Testteilnehmer aus der Gruppe der Sonstigen darauf verweisen, dass ihnen dieser Fehler geläufig ist oder diesen bereits selbst gemacht haben. Ein Beispiel hierfür ist diese Antwort: *„Das erste Element im Array hat den Index 1 - den Fehler mache ich manchmal auch noch. ;)“*. Ob die Studierenden bei ihrer Antwort ihre eigenen Fehlvorstellungen heranziehen, bleibt offen, da dies nicht explizit in den Antworten erwähnt wurde.

Bei diesem Item wurden vergleichsweise wenige falsche Antworten abgegeben, die sich wiederum inhaltlich mehrheitlich auf Fehler in den einzelnen Schritten beziehen. Worin letztere jedoch genau bestehen, wurde nie genauer spezifiziert. Ein Beispiel dafür ist die Antwort eines Studierenden: *„Im letzten Schritt wurde die +2 als anderer Wert interpretiert“*. Welche Annahmen hier getroffen wurden und wie diese letztendlich zu einer falschen Antwort führen, ist nicht nachvollziehbar.

Item 4 - Datenstrukturen, Queue

Wie bereits in Kapitel 5 beschrieben, liegt die besondere Herausforderung dieses Items darin, nicht die zunächst offensichtliche Antwort auszuwählen, sondern zwischen den drei vorgegebenen Möglichkeiten abzuwägen und diese entsprechend zu begründen. Vor Durchführung der Erhebung liess sich nicht ausschließen, dass für eine der beiden anderen Antwortoptionen nicht erwartete, aber korrekte Antworten abgegeben werden würden. Dies war jedoch nicht der Fall.

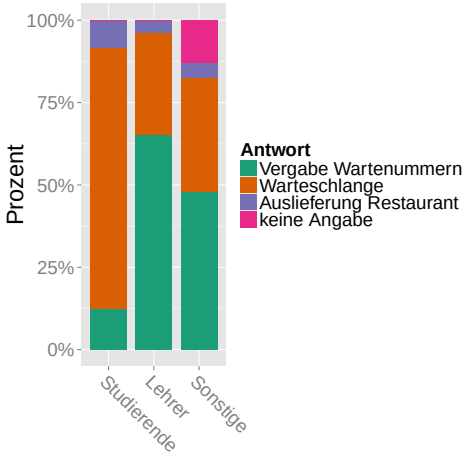


Abbildung 6.6.: Item Datenstrukturen A - DS1

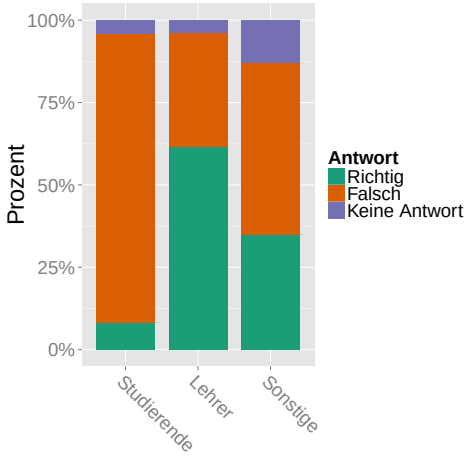


Abbildung 6.7.: Item Datenstrukturen B - DS2

Dieses Item hatte die geringste Quote an richtigen Antworten von Studierenden (12.50 % für den ersten Teil bzw. 8.33 % für den zweiten). Wie in Abbildung 6.8 zu erkennen ist, entschied sich die Mehrheit (79.17 %) für die Antwortoption „Warteschlange im Supermarkt“. Begründet wurde dies oftmals mit der direkten Übersetzung des Namens, z. B.: *„Wie der Name schon sagt entspricht die Queue einer Warteschlange: Objekte werden angeordnet und der Reihe nach abgearbeitet.“*. Auch wurde mehrmals darauf hingewiesen, dass die Warteschlange an einer Kasse ein häufig verwendetes Beispiel ist: *„Der Supermarkt ist eigentlich das klassische Beispiel, ich kann darin auch nichts falsches erkennen“*. Hier wurde somit offensichtlich nicht anhand der eigenen Einschätzung ausgewählt, sondern auf Basis von erworbenen Wissen: der Befragte hat dieses Beispiel bereits ausserhalb des Tests kennengelernt und geht so davon aus, dass es geeignet ist.

Item 5 - Logik

Die Ergebnisse der drei Items zum Thema Logik zeigen eine große Lücke zwischen den Gruppen der Studierenden und Lehrern und der Gruppe der Sonstigen. Während in letzterer 65.22 % der Probanden eine richtige Antwort gaben, sind dies bei den Studierenden lediglich 20.83 % und bei den Lehrern 26.92 %. Die Verteilung der Antworten, insbesondere bei Teil B und C, lässt vermuten, dass geraten wurde bzw. zunächst Antworten per Ausschlussverfahren ausgeschlossen wurden.

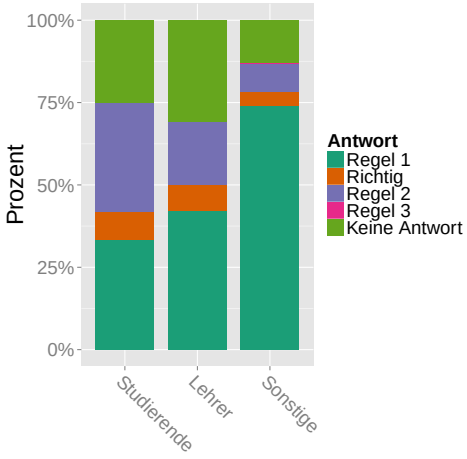


Abbildung 6.8.: Item Logik A - L1

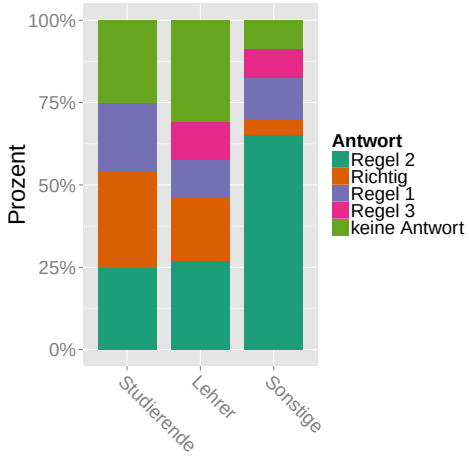


Abbildung 6.9.: Item Logik B - L2

Dieses Item ist das einzige Item, das die Reliabilität des Tests (Cronbachs α) nicht senkt, wenn es entfernt wird. Durch diese Ergebnisse wird deutlich, dass dieses Item im Gegensatz zu den anderen Items des Testinstruments nicht das Wissen über Fehlvorstellungen misst. Aufgrund der geschlossenen Form des Items kann im Nachgang leider nicht nachvollzogen werden, warum der Anteil der falschen Antwort insbesondere bei den Studierenden und den Lehrern hoch war. Falls bei einem weiteren Einsatz des Instruments an diesem Item festgehalten werden soll und es dementsprechend angepasst werden soll, müsste dies zunächst beantwortet werden.

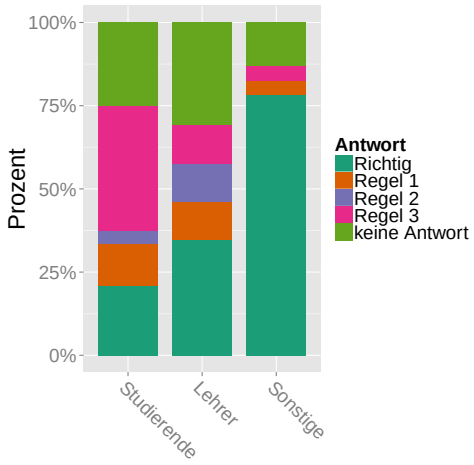


Abbildung 6.10.: Item Logik C - L3

Item 6 - Programmierung, Code

Im Gegensatz zu den zuvor präsentierten zweiteiligen Items, enthält das Item zum Thema Programmierung kein vorangestelltes fachwissenschaftliches Item, sondern besteht aus zwei Teilen, die sich jeweils auf Fehlvorstellungen beziehen. Wie bereits in Kapitel 5 erläutert, können die Antwortmöglichkeiten auf dieses Item als eindeutig verstanden werden. Die Parallele zur Mathematik bei der Zuweisung zur Variablen und die Annahme, dass beide Teile der if-else-Anweisung ausgeführt werden, stellen somit die Musterlösung dar. Dies bestätigte sich bei der Auswertung des Tests. Die als falsch bewerteten Antworten nahmen mehrheitlich Bezug darauf, dass im ersten Teil der Aufgabe ein Syntaxfehler vorhanden ist und im zweiten Teil der print-Befehl versehentlich in die Klammer des else-Parts geschrieben wurde. Sie gehen damit nicht auf tatsächliche Fehlvorstellungen ein.

Zudem zeigte sich in einigen Antworten auch die Benutzung falscher Fachtermini. Der Testteilnehmer aus der Gruppe der Studierenden erwähnt hier die oft dokumentierte, aber offensichtlich nie aussterbende „If-Schleife“: „*wenn die zahl gerade ist bricht die schleife ab und springt gar nicht erst in das if*“.

Der erste Teil dieses Items hat mit 88.46 % die höchste Quote richtiger Antworten alles Items bei der Gruppe der Lehrer. Dies legt die Vermutung nahe, die sich bei der Durchführung der Pretests mit diesem Item ergab, dass es sich hierbei um

in der Praxis häufig auftretende Fehler handelt, die dementsprechend auch den Lehrern wohlbekannt sind.

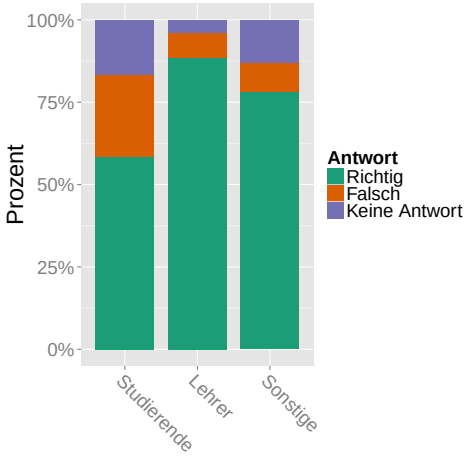


Abbildung 6.11.: Item Programmierung A - PR1

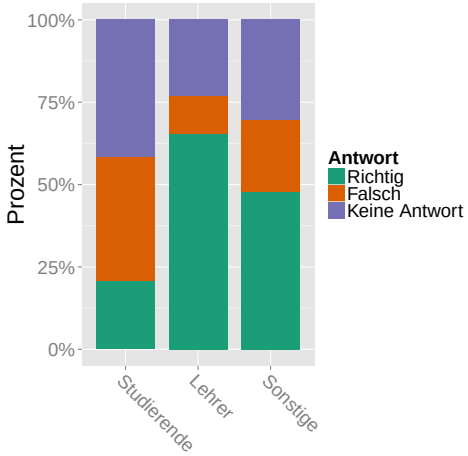


Abbildung 6.12.: Item Programmierung B - PR2

Item 7 - Systeme, Analogien

Wie in der Vorbewertung (vgl. Kapitel 5) bereits vermutet, stellt dieses Item mit seiner offenen Fragestellung eine besondere Herausforderung dar. Dies wird besonders dadurch deutlich, dass der Anteil der fehlenden Antworten (d. h. ein leeres Antwortfeld) bei insgesamt 63.01 % liegt, bei der Gruppe der Sonstigen sogar bei 73.91 %.

Dieses Item war eindeutig das Item, bei dem die Bewertung der Antworten sich am schwierigsten gestaltete. Dies ist zum einen darauf zurückzuführen, dass eine Eingrenzung mit Hilfe einer Musterlösung oder beispielhaften Lösungen nicht möglich ist. Zum anderen waren viele der Antworten sehr unpräzise und ließen eine genauere Erläuterung vermissen. Ein Beispiel hierfür ist die folgende Antwort eines Lehrers: *„Eis und Wasser können nicht verloren gehen, der Inhalt von Arbeitsspeicher allerdings schon“*. Wie andere Antworten, bezieht sich diese vermutlich darauf, dass Arbeitsspeicher im Regelfall flüchtig ist, d. h. die gespeicherten Informationen sind nicht mehr vorhanden, wenn die Stromversorgung beendet wird. Inwieweit hier allerdings der Vergleich zum Eis hergestellt werden soll, wird durch den Begriff

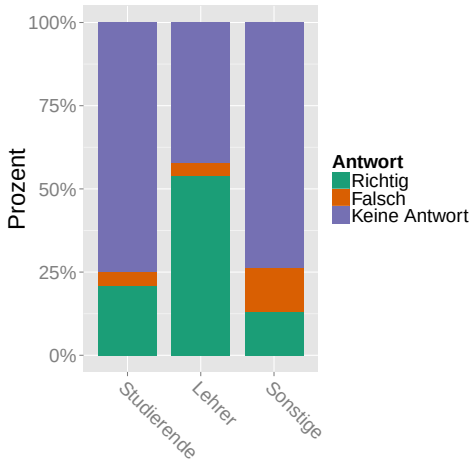


Abbildung 6.13.: Item Systeme - Sys

„verloren gehen“ nicht deutlich und muss interpretiert werden, was wiederum eine objektive Bewertung verhindert. Trotz der hohen Signifikanz sollte daher hinterfragt werden, ob sich das Item in der aktuellen Form für das Instrument eignet. Gegebenenfalls könnte eine Überarbeitung hilfreich sein, bei der bestimmte Kriterien des Arbeitsspeichers genannt werden (z. B. Lese- und Schreibvorgänge) und der Befragte die Eignung der Analogie hinsichtlich dieser Kriterien überprüfen muss.

Die überwiegende Mehrheit der Antworten lässt sich den folgenden Kategorien zuordnen:

- Überschreiben: *„Ein Eisblock kann nicht überschrieben werden, d.h. man kann ein Element des Eis nicht problemlos durch ein anderes ersetzen. Somit ist es auch schwierig zu verstehen wie dies bei Speicher funktioniert.“*
- Flüchtigkeit: *„Arbeitsspeicher ist flüchtig, sobald die Stromversorgung weg ist, wird er gelöscht. Auch wenn Eis aufgetaut wird, ist es nicht weg, sondern kann sogar wieder gefroren werden.“*
- Aufbau und Konsistenz: *„Eis entspricht physisch einfach in keinster Weise dem Aufbau der Hardware von Arbeitsspeicher. Dieser gleicht im Regelfall einer Matrix. Bei Eis ist das nicht vorstellbar.“*
- Geschwindigkeit: *„Der Arbeitsspeicher ist sehr schnell. Informationen werden in Sekundenbruchteilen geschrieben und gelesen. Eis in Wasser umzuwandeln ist vermutlich auch mit den besten Werkzeugen langsamer.“*

Bei der Auswertung fiel ausserdem auf, dass die Antworten hier tendenziell sehr stichwortartig geschrieben wurden und damit eher wie eine Stichwortsammlung oder eine Vermutung formuliert waren. Ein Beispiel dafür ist diese Antwort eines Studierenden: „*Es gibt da wohl immer etwas was nicht passt... hier finde ich das Beispiel sehr absrtakt, da speicher ja grundsätzlich eine hardware ist, die immer bestehen bleibt. Eis ist irgendwann einfach weg, verdunstet...*“.

6.5. Weitere Aspekte

In der Erhebung wurde im Rahmen der Abfrage der demographischen Daten auch erhoben, wie viele Jahre Lehrerfahrung die Probanden haben. Diese schlüsselt sich in 5 Zeiträume auf (keine, weniger als 1 Jahr, 1-5 Jahre, 6-10 Jahre und mehr als 10 Jahre). Die Daten können Aufschluss darüber geben, ob ein Zusammenhang zwischen der Lehrerfahrung in Jahren und dem Abschneiden im Test existiert. In Abbildung 6.14 ist dies veranschaulicht, indem die Durchschnittswerte für die einzelnen Gruppen als Balkendiagramm dargestellt werden. Daraus lässt sich erkennen, dass die durchschnittliche Punktzahl höher ist, wenn die Lehrerfahrung größer ist.

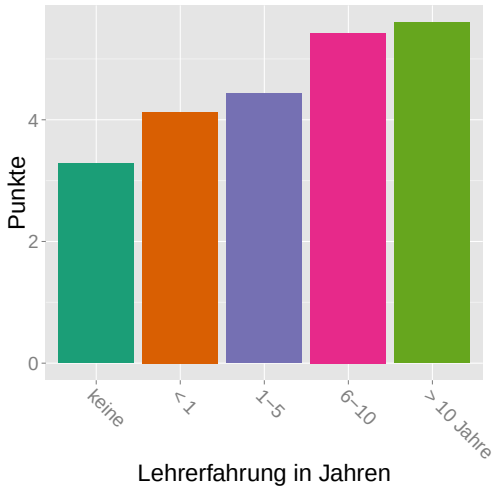


Abbildung 6.14.: Durchschnittliche Punktzahl nach Lehrerfahrung

In Tabelle 6.4 sind diese Daten noch einmal übersichtlich dargestellt und aufgeschlüsselt. Der Korrelationskoeffizient liegt bei $r_s = 0.44$, es liegt somit ein

schwacher Zusammenhang vor. Aus der Tabelle wird allerdings auch deutlich, dass die Größe der einzelnen Gruppe sehr unterschiedlich ist. Während 39 Probanden keine Lehrerfahrung haben, liegt diese nur bei 5 Personen bei mehr als 10 Jahren. Diese Stichprobengrößen sind zu klein, um zuverlässige Rückschlüsse auf die Gesamtpopulation ziehen zu können.

Lehrerfahrung	durchschn. Punkte	Standardabweichung σ	Anzahl
keine	3.28	1.79	39
$<$ 1 Jahr	4.12	2.36	8
1-5 Jahre	4.75	2.31	14
6-10 Jahre	5.43	1.27	7
$>$ 10 Jahre	5.60	1.67	5

Tabelle 6.4.: Auswertung nach Lehrerfahrung

Ähnliche Ergebnisse zeigen sich auch bei der Betrachtung der durchschnittlichen Punktzahl und der didaktischen Ausbildung. So lag der Mittelwert bei den Absolventen eines Lehramtstudiums Informatik bei $\bar{x} = 4.88$, während er bei den Probanden ohne entsprechenden Abschluss bei $\bar{x} = 3.45$ lag.

6.6. Zusammenfassung

Die Studie hat gezeigt, dass die Gruppe der Lehrer bei der Beantwortung der Testitems die besten Ergebnisse erzielt hat. Das Gesamtergebnis ist dabei statistisch hoch signifikant und die Mehrheit der Ergebnisse der einzelnen Items des Instruments war signifikant bis hoch signifikant. Aufgrund der Tatsache, dass dies der erste Einsatz des Testinstruments ist und die Items zuvor nur in informellen Pretests überprüft werden, ist dies ein sehr erfreuliches Ergebnis.

In der Auswertung zeigt sich, dass es zwischen dem Lehramtsstudium und der Tätigkeit als Lehrer einen starken Wissenszuwachs beim Wissen über Fehlvorstellungen und der Fähigkeit, Fehlvorstellungen zu identifizieren, gibt. Die zwar etwas schlechteren, doch gleichermaßen guten Ergebnisse der sonstigen Teilnehmer, die weder eine didaktische Ausbildung absolviert, noch selbst unterrichtet haben, zeigen gleichermaßen, dass dieses Wissen auch durch eine rein informatische Ausbildung und Praxiserfahrung gewonnen werden kann. Dies bestätigt Ergebnisse aus vergleichbaren Studien wie z.B. der COACTIV-Studie (Krauss u. a., 2008a). Hier erzielten Studierende des Hauptfachs Mathematik höhere Ergebnisse im Gebiet

der Fehlvorstellungen und Schwierigkeiten von Schülern als Gymnasiallehrer. Auf Basis dieser Datenlage darauf zu schließen, dass eine fachdidaktische Ausbildung keine, bzw. nur eine geringe Wirkung auf diese Kompetenzen hat, wäre allerdings fahrlässig. So waren in beiden Studien (der COACTIV-Studie und der Studie der vorliegenden Arbeit) die Items auf die theoretische Analyse von Fehlvorstellungen ausgelegt. Wie dies in der Praxis funktioniert, bleibt unbeachtet. Auch war die Teilnahme an den Studien freiwillig und die Vermutung liegt nahe, dass bei allen Teilnehmern ein grundsätzliches Interesse am Thema bestand.

Bei der Auswertung der offenen Antworten wurde deutlich, dass das Instrument in gewisser Weise rekursiv konstruiert ist, was auch einen großen Vorteil für zukünftige Erweiterungen und Überarbeitungen darstellt. In Antworten, in denen die Testteilnehmer bekannte und identifizierte Fehlvorstellungen beschreiben sollten, versteckten sich mehrfach bei den Probanden selbst vorhandene Fehlvorstellungen. Dies bestätigte auch die Entscheidung, Items möglichst auf einem fachwissenschaftlich geringen bzw. für die Beantwortung unbedeutenden Level zu entwerfen oder, wenn dies nicht möglich ist, das Wissen des Probanden vorab zu überprüfen.

Eine weitere Beobachtung, die die Autorin dieser Arbeit bei der Auswertung der Erhebung gemacht hat, die aber aufgrund der geringen Anzahl der Probanden nur schwer belegbar ist, ist die Perspektive aus der die offenen Antworten geschrieben werden. Hier zeigt sich, dass die Teilnehmer aus der Gruppe der Lehrer sehr oft direkt auf die Person eingehen, deren Fehlvorstellung in der Beschreibung des Items erläutert wurde. Dies ist aufgrund der Fragestellungen nicht unüblich. Auffallend ist jedoch, dass die Antworten der beiden anderen Gruppen auf die Perspektive bzw. Erfahrung der Antwortenden eingehen, indem sie z. B. erwähnen, dass sie bestimmte Fehlvorstellungen selbst hatten oder diese bei anderen beobachtet haben.

6.7. Reflexion zur Eignung des Testinstruments und der einzelnen Items

6.7.1. Itemformate

Hinsichtlich der Formulierung und Konzeption der Items haben sich die in Kapitel 5 genannten testtheoretischen Kriterien und Überlegungen bestätigt.

Die zuvor genannte Problematik dass Antworten auf offene Fragen nicht problemlos als „richtig“ oder „falsch“ klassifiziert werden können, hat sich bei dieser Erhebung

nicht bestätigt. Vereinzelt fehlten Begründungen oder Erklärungen zu Antworten auf offene Fragen - wenn diese vorhanden waren, ließen sich jedoch überwiegend unstrittig zuordnen.

Bemerkenswert ist in diesem Kontext, dass die unterschiedlichen Fragestellungen bzw. die Fokussierung der Fragen zu unterschiedlichen Ergebnissen führt. So zeigte sich dass zu Item 7, in dem offen nach möglichen Fehlvorstellungen zu einer eher unbekannten Analogie gefragt wurde, die Antworten deutlich breiter aufgestellt waren, während sich diese bei Item 4, bei dem häufig verwendete Analogien betrachtet werden sollten, nur sehr wenige Kategorien einordnen ließen. Dieser Unterschied kann allerdings auch dazu führen, dass Antworten weniger leicht zu bewerten sind. Dementsprechend war dies bei Item 7, im Vergleich zu den anderen Aufgaben dieses Testinstruments, besonders herausfordernd.

Ein Nachteil der offenen Fragen, insbesondere bei der Verwendung eines Testinstruments, ist allerdings auch, dass hier nicht immer offensichtlich ist, ob eine fehlende Antwort tatsächlich mit fehlendem Wissen gleichzusetzen ist, oder ggf. auch durch Zeitmangel oder andere Faktoren, ausgelassen wurde. Dies zeigt sich besonders in Item 7, bei dem der Anteil der fehlenden Antworten bei insgesamt 63.01 % lag und welches nach Einschätzung der Autorin auch das Item war, bei dem die Antwort am wenigsten naheliegend war.

6.7.2. Auswahl der Themen

Die Ergebnisse der fachwissenschaftlichen Teil-Items belegt, dass das Fachwissen bei der Beantwortung der Items elementar ist. Insbesondere die Ergebnisse von Item 5 legen nahe, dass fehlerhafte Antworten überwiegend auf fehlendem Fachwissen basieren. Dies bedeutet gleichermaßen auch, dass dieses Item nicht bzw. nicht ausschließlich das Wissen über Fehlvorstellungen misst.

7. Zusammenfassung, Fazit und Ausblick

In der vorliegenden Arbeit wurde ein Testinstrument entwickelt, mit dem das Wissen über Fehlvorstellungen in der Informatik gemessen werden kann. Im Folgenden sollen die in Kapitel 1 formulierten Forschungsziele noch einmal im Überblick genannt und ihre Erreichung diskutiert und veranschaulicht werden. Dies schließt ein Fazit zu den einzelnen Zielen ein, in dem die zuvor beschriebenen Forschungsansätze mit den Ergebnissen verglichen werden sollen, um ihr Erreichen zu überprüfen. Im darauf folgenden Abschnitt 7.2 erfolgt ein Ausblick, welche Aspekte sich im Rahmen des Forschungsvorhaben eröffnet haben, die in zukünftigen Forschungsansätzen aufgenommen und weiter betrachtet werden könnten.

7.1. Zusammenfassung und Fazit

F1: Analyse und Bewertung des aktuellen Forschungsstandes zu Fehlvorstellungen in der Informatik

Nachdem in Kapitel 2 zunächst eine allgemeine Erläuterung und Beschreibung der Forschung zu Fehlvorstellungen stattfand und die in dieser Arbeit verwandten Definitionen vorgestellt wurden, wurden diese Ergebnisse in Kapitel 3 auf das Gebiet der Informatik übertragen und dort publizierte Forschungsergebnisse vorgestellt und analysiert. Ein entscheidender Aspekt in diesem Kontext war, Forschungsansätze zu vereinen und abzugrenzen. So war es möglich, diese zu strukturieren und zu sortieren, aber auch kritisch zu beleuchten und zu hinterfragen.

Positiv zu bemerken ist, dass vielversprechende Ergebnisse zu Fehlvorstellungen in der Informatik existieren und bereits eine Vielzahl an Fehlvorstellungen dokumentiert wurde. Es wurde bei der Analyse jedoch gleichzeitig immer wieder

deutlich, dass die Forschung noch jung ist. Dies war zum einen in der Konzeption empirischer Erhebungen offensichtlich. Studien wurden oftmals mit einer sehr kleinen Stichprobengröße durchgeführt, die wenig Möglichkeit gibt, zuverlässige Schlussfolgerungen aus den Ergebnissen zu ziehen. Gleichzeitig beschränken sich nicht wenige Studien auf eine sehr enge Auswahl der Testteilnehmer, was sich insbesondere auf die jeweilige Ausbildung bezieht. So sind Messungen von Studierenden in einer Vorlesung oder Schülern in einer Klasse häufig. Dies ist gerade beim Thema Fehlvorstellungen fragwürdig, da die Lehre einen großen Einfluss auf das Entstehen von Fehlvorstellungen hat.

Ein weiteres Ergebnis der Analyse ist, dass bei weitem nicht alle Themenbereiche in Forschungsvorhaben betrachtet wurden. Ganz im Gegenteil existieren große Lücken wie z. B. im Gebiet der theoretischen Informatik, wo vorhandene Forschungsergebnisse überwiegend aus der Mathematik stammen. Allerdings besteht eine Häufung an Studien in der Objektorientierung, die jedoch kaum verzahnt sind. Aus diesem Grund existieren nur wenige einheitliche Messinstrumente, mit denen Fehlvorstellungen gemessen werden könnten.

Leider wurde mehrfach beim Vergleich einzelner Forschungsansätze und Forschungsergebnisse deutlich, dass diese nicht frei von Widersprüchen sind. So werden einzelne Fehlvorstellungen mit sehr unterschiedlicher Häufigkeit beobachtet und gegensätzliche Empfehlungen zur Vermeidung gegeben. Dies mag in manchen Fällen angemessen sein, da sich der Kontext im Detail unterscheidet, oftmals fehlt allerdings eine klare Linie, sowie der Rückgriff auf zuvor publizierte Ergebnisse oder ein Vergleich mit diesen.

Nichtsdestotrotz konnte im Rahmen dieser Arbeit eine umfangreiche Sammlung an Forschungsansätzen und ihren Ergebnissen erstellt werden, die besonders der Strukturierung und Orientierung dient. Die darin gesammelten Ergebnisse konnten gewinnbringend im weiteren Verlauf der Arbeit, insbesondere bei der Messung des Wissens über Fehlvorstellungen, eingesetzt werden.

F2: Analyse und Erhebung relevanter Konzepte und der Häufigkeit von Fehlvorstellungen in der Informatik

In Kapitel 4 wurden einleitend verschiedene Methoden und Kriterien zur Identifikation relevanter Konzepte in der Informatik vorgestellt und beurteilt. Dabei zeigte sich, dass leider keines davon ohne Einschränkung für das Ziel dieser Arbeit zu adaptieren ist. Dies ist zunächst einmal der Zielgruppe zuzuschreiben, da sich nur

wenige der ermittelten Konzeptsammlungen auf die Eingangsphase der Informatikausbildung beschränken. Wenn dies der Fall ist, sind die Konzepte wiederum sehr allgemein gehalten und Definitionen oder Hintergrundinformationen fehlen.

Hinzu kommt, dass nur wenige Erkenntnisse über die Häufigkeit von Fehlvorstellungen oder über die Wahrscheinlichkeit der Entstehung von Fehlvorstellungen zu einzelnen Konzepten vorhanden sind. Wenn dies empirisch ermittelt oder gemessen wurde, beschränkte sich dies auf einzelne Themengebiete oder sogar nur auf einzelne Themen. Zudem war auch dann die Vollständigkeit der Ergebnisse fragwürdig.

Es war daher nicht möglich, geeignete Ansätze zur Identifikation relevanter Konzepte und zur Häufigkeit von Fehlvorstellungen zu verbinden, um daraus eine Sammlung der wichtigsten und für Fehlvorstellungen anfälligsten Konzepte zu erhalten. Daher wurde im Kontext dieser Arbeit eine entsprechende Erhebung durchgeführt.

Zu diesem Zweck wurde zunächst eine Auswahl an möglichen Konzepten generiert, indem eine Häufigkeitsanalyse der Inhalte der Einführungskurse des ACM/IEEE-Curriculum durchgeführt wurde. Diese wurden daraufhin in einer Online-Umfrage von Universitätsprofessoren nach den Kriterien der Fundamentalen Ideen und der Häufigkeit von Fehlvorstellungen bewertet.

Die Entscheidung zur Verwendung der Kriterien der Fundamentalen Ideen als Indikator fiel auf Basis mehrerer Erwägungen. Sie stellen einen der wenigen Ansätze dar, der in einer einzelnen Iteration durchführbar ist. Die Möglichkeit, auf diesem Weg valide Ergebnisse zu erhalten, wurde in anderen Studien bereits gezeigt. Gleichzeitig lässt sich durch eine Bewertung der Konzepte durch eine Vielzahl an Experten einer der am häufigsten erwähnten Kritikpunkte, nämlich die Objektivität, deutlich minimieren. Zwar liegt nahe, dass durch die Tätigkeit der Teilnehmer als Universitätsprofessoren gewisse Einflüsse oder Tendenzen bei der Bewertung nicht vermeidbar sind, doch haben andere Studien gezeigt, dass dies nicht zwangsläufig von Nachteil ist (vgl. Kapitel 4).

Die Bewertungen wurden mit Hilfe einer Cluster-Analyse zusammengefasst, so dass Konzepte mit ähnlichen Bewertungsschemata der einzelnen Kriterien der fundamentalen Ideen zu einem Cluster gruppiert und ihnen die durchschnittliche Bewertung der Häufigkeit von Fehlvorstellungen zugeordnet wurde. Bemerkenswert dabei war, dass die Mehrheit der Konzepte, die hohe Bewertungen bei den Kriterien der Fundamentalen Ideen erhalten hatten (die somit nicht als Fundamentale Idee angesehen werden können), auch sehr hohe Bewertungen bei der Wahrscheinlichkeit von Fehlvorstellungen erhalten habe. Dies führt zu der Interpretation, dass Konzepte, die nicht fundamental für die Informatikausbildung sind, auch meist

mit deutlich mehr Fehlvorstellungen verbunden sind. Das ist für diese Arbeit eine herausfordernde Konstellation, da Konzepte identifiziert werden sollten, die sowohl relevant sind, als auch häufig mit Fehlvorstellungen in Verbindung gebracht werden.

Dies wurde abschließend in der Form gelöst, dass diejenigen Konzepte ausgewählt wurden, die weder in der einen, noch in der anderen Kategorie Bewertungen im extremen Bereich hatten, d. h. insbesondere keine Konzepte, bei denen die Häufigkeit von Fehlvorstellungen als sehr gering eingeschätzt wurde oder für die die Kriterien der Fundamentalen Ideen als nicht erfüllt bewertet wurden. Zudem wurden einzelne Konzepte zusammengefasst, so dass letztendlich sechs Themenkomplexe identifiziert werden konnten, zu denen im folgenden Schritt Items generiert wurden.

F3: Entwicklung und Einsatz eines Testinstruments zur Messung des Wissens über Fehlvorstellungen in der Informatik

Basierend auf den identifizierten Konzepten erfolgte ein Überblick über die für die jeweiligen Bereiche relevanten Fehlvorstellungen. Dieser diente insbesondere dazu, einen Einblick in die Thematik zu geben und zur Quelle für die Fehlvorstellung hinzuführen, die anschließend als Testitem umgesetzt wurde.

Es erwies sich als weitgehend problemlos, dokumentierte Fehlvorstellungen zu den vorab identifizierten Konzepten in Publikationen zu Fehlvorstellungen zu finden. Auch konnten diese zuverlässig in die in dieser Arbeit benötigte Form, d. h. zur Messung des Wissens über Fehlvorstellungen, übertragen werden. Ein Aspekt, der hier besonders beachtet werden musste, war die Form der Items. Hier musste eine Entscheidung zwischen einem offenen oder einem geschlossenen Frageformat gefunden werden. In den meisten Fällen musste davon ausgegangen werden, dass die Vorgabe von Antworten im Multiple-Choice-Format nicht zielführend wäre. Zum einen sind Antworten meist offensichtlich, wenn sie vorgegeben werden. Die deutlich größere Herausforderung besteht darin, die Fehlvorstellung überhaupt zu erkennen. Zum anderen würde so der wünschenswerte Nebeneffekt ausgeschlossen, dass sich in den Antworten auch neue und bislang unbekannte Fehlvorstellungen wiederfinden, die bei der Konzeption des Testitems nicht bekannt und in den zugrundeliegenden Publikationen auch nicht dokumentiert wurden.

Die zuvor genannten Vorteile eines offenen Frageformats führen allerdings gleichermaßen zu einem Nachteil: Antworten können gegebenenfalls schwer zu bewerten sein, wenn nicht dokumentierte, eventuell unwahrscheinlich wirkende Fehlvorstellungen beschrieben werden. Diese Sorge hat sich allerdings bei der Auswertung der Antworten als unbegründet erwiesen. Wie oben bereits angedeutet, lag die

größte Herausforderung für die Probanden offensichtlich darin, überhaupt plausible Fehlvorstellungen zu erkennen, die fachlich angemessen sind. Die überwiegende Mehrheit der mit 0 Punkten bewerteten Lösungen bestand entweder aus nicht beantworteten Fragen oder aus Antworten, die selbst eine Fehlvorstellung des Testteilnehmers vermuten ließen. Um diese bei der Bewertung einfacher identifizieren zu können, wurde mehreren Items eine fachwissenschaftliche Frage vorangestellt, die zunächst richtig beantwortet werden musste. Dies erwies sich als sehr wertvoll, da in mehreren Fällen nicht ersichtlich war, in welche „Richtung“ die Testteilnehmer argumentierten. So wurde im Item zur Effizienz von Algorithmen die „Anzahl der Variablen“ als Argument genannt, so dass nicht deutlich wurde, ob hier auf die höhere oder niedrigere Anzahl eines der beiden Algorithmen Bezug genommen wurde. Durch die vorgestellte Frage konnte dies eindeutig geschlossen werden.

Die mit Hilfe von LimeSurvey implementierte Umfrage wurde an Studierende im Lehramt Informatik, Informatik Lehrer und Absolventen im Fach Informatik (d.h. Personen, die in Berufen im Gebiet der Informatik arbeiten) versendet. Diese wurden zusätzlich gebeten, die Einladung an ihnen bekannte Personen, die zu dieser Zielgruppe gehören, weiterzuleiten. Insgesamt konnten die Antworten von 73 Probanden in die Auswertung einbezogen werden, so dass die aus statistischer Sicht benötigte Stichprobengröße erfüllt werden konnte.

Zusammenfassend konnten aus der Studie überzeugende und wertvolle Ergebnisse gewonnen werden. Die Gruppe der Lehrer schnitt dabei mit Abstand zu der Gruppe der Sonstigen am besten ab. Der hier vorhandene Unterschied ist, ebenso wie der Abstand zur Gruppe der Studierenden, höchst signifikant. Dies trifft auch für die Mehrzahl der einzelnen Items zu.

Einige Abstriche mussten beim Item zum Thema Logik gemacht werden. Schon bei der Konzeption entstanden Zweifel, inwieweit die Thematik und die daraus zu generierende Aufgabe für eine Erhebung in dieser Form geeignet ist. Aus diesem Grund wurde es auch mehrfach überarbeitet und in eine Form überführt, bei der es nicht nötig ist, Fachbegriffe aus der Logik zu kennen. Die Ergebnisse der Vortests zeigten, dass dieses Item zwar das schwierigste ist, aber durchaus richtig beantwortet wird. Umso überraschender waren die Ergebnisse sehr schlecht und besondere bei der Gruppe der Studierenden auffallend gemischt, so dass sich vermuten lässt, dass geraten wurde. Dieses Item ist auch das einzige Item, das die interne Konsistenz (Cronbachs α) senkt und somit bei einer erneuten Erhebung überarbeitet oder ausgeschlossen werden sollte. Des Weiteren ergaben sich aus den Ergebnissen verschiedene Anschlussfragen, die aufgrund nicht erhobener demographischer Daten nicht beantworten werden konnten (vgl. Abschnitt 7.2).

Das entstandene Instrument stellt dennoch in seiner Gesamtheit einen wertvollen Beitrag für die Forschung zu Fehlvorstellungen dar. Bisher existierten keine Messinstrumente zur Messung des Wissens über Fehlvorstellungen, die erprobt und ausgewertet wurden.

7.2. Ausblick

Die vorliegende Arbeit stellt neben der empirischen Ermittlung von relevanten und häufig mit Fehlvorstellungen verbundenen Konzepten ein Instrument vor, mit dem das Wissen über Fehlvorstellungen gemessen und verglichen werden kann. Um das Testinstrument weiter zu verwenden, zu erweitern und verbessern, sowie den Forschungsprozess fortzusetzen, existieren mindestens zwei Möglichkeiten:

Erhebung weiterer demographischer Daten: Die erste Möglichkeit liegt in einer feineren Erhebung von Daten. In der im Rahmen dieser Arbeit durchgeführten Studie wurden die Ergebnisse lediglich hinsichtlich der Zugehörigkeit zu einer der drei eingeladenen Gruppen an Teilnehmern ausgewertet. Eine ausführlichere Auswertung war zum einen aufgrund der nur begrenzt erhobenen demographischen Daten nicht möglich. Zum anderen war die Stichprobe zu klein, als dass man weitere Merkmale zuverlässig statistisch hätte auswerten können. So wäre es beispielsweise interessant, in einer zukünftigen Studie auch den Studienstand der Studierenden zu erheben, um so Rückschlüsse auf den genaueren Zeitrahmen der Entwicklung des Wissens ziehen zu können. Ideal wäre es dabei, weitere Gruppen einzubeziehen. Dies könnte einhergehen mit der Durchführung eines Pre- und Posttests, um z. B. den Erfolg des Praxissemesters bei Lehramtsstudierenden zu messen.

Erweiterung und Verfeinerung des Testinstruments: Die Auswertung in Kapitel 6 hat gezeigt, dass die Items zum Thema Logik nicht für die Messung des Wissens über Fehlvorstellung geeignet sind. Diese sollten somit ersetzt oder umformuliert werden. Auch fand schon bei der Konstruktion der Items eine gewisse thematische Einschränkung statt, die für die in dieser Arbeit durchgeführte Studie nötig war, um den Umfang des Testinstruments in einem akzeptabel Rahmen zu halten. Eine Erweiterung ist aber naheliegend. Aufgrund der guten Dokumentation von Fehlvorstellungen in einigen Themengebieten (z. B. den Datenstrukturen) ist dies mit eher geringem Aufwand umsetzbar und würde sich gleichzeitig positiv auf die interne Konsistenz auswirken. Bei einer entsprechenden Stichprobengröße wäre

es möglich, eine größere Anzahl Items beantworten zu lassen und jeden Teilnehmer nur einen randomisierten Anteil an Testitems beantworten zu lassen, um die Beantwortungszeit klein zu halten. Auch bietet es sich an, die Items zu übersetzen und international einzusetzen. Da keinerlei sprachspezifischen Elemente im Instrument vorhanden sind, ist dies ohne zusätzliche Anpassungen möglich.

Es schließen sich weitere Fragen zu Nebenergebnissen dieser Arbeit an, deren nähere Betrachtung sicherlich von Interesse ist. Dies gilt beispielsweise für das schlechte Abschneiden der Objektorientierung bei der Bewertung der Kriterien der Fundamentalen Ideen (vgl. Kapitel 4). Hier wäre es insbesondere interessant, nach Gründen für die entsprechenden Bewertungen zu fragen oder aber auch, die Studie mit einer weiteren Personengruppe durchzuführen.

Appendix

Anhang A.

Sammlung von Quellen

Im folgenden wird ein Überblick über Publikationen zu Fehlvorstellungen gegeben, die im Rahmen dieser Arbeit analysiert bzw. verwendet wurden. Da viele thematisch nicht eindeutig zu einer Kategorie zugeordnet werden können, wurde der jeweilige Schwerpunkt ausgewählt oder, falls es mehrere Schwerpunkte gab, fand eine Zuordnung zu beiden Kategorien statt.

A.1. Spezifische Fehlvorstellungen

A.1.1. Algorithmen

Titel	Quelle
Identification and removal of misconceptions about greedy algorithms assisted by an interactive system	Iturbide (2012)
Hunting High and Low: Instruments to Detect Misconceptions Related to Algorithms and Data Structures	Paul u. Vahrenhold (2013)
Detecting and Understanding Students' Misconceptions Related to Algorithms and Data Structures	Danielsiek u. a. (2012)
Commonsense Computing (episode 5): Algorithm Efficiency and Balloon Testing	Mccartney u. a. (2009)

A comparison of the misconceptions about the time-efficiency of algorithms by various profiles of computer-programming students	Özdener (2008)
Metaphors and Analogies for Teaching Algorithms	Forišek u. Steinová (2012)
Commonsense computing: what students know before we teach (episode 1: sorting)	Simon u. a. (2006)
The efficiency of algorithms - Misconceptions	Gal-Ezer u. Zur (2003)
Developing and validating test items for first-year computer science courses	Vahrenhold u. Paul (2014)

Tabelle A.1.: Publikationen zu Fehlvorstellungen zum Thema Algorithmen

A.1.2. Betriebssysteme

Titel	Quelle
Exploring misconceptions of operating systems in an online course	Pamplona u. a. (2013)
Developing a pre- and post-course concept inventory to gauge operating systems learning	Webb u. Taylor (2014)

Tabelle A.2.: Publikationen zu Fehlvorstellungen zum Thema Betriebssysteme

A.1.3. Datenstrukturen

Titel	Quelle
Hunting High and Low: Instruments to Detect Misconceptions Related to Algorithms and Data Structures	Paul u. Vahrenhold (2013)
Detecting and Understanding Students' Misconceptions Related to Algorithms and Data Structures	Danielsiek u. a. (2012)
Misconceptions and Concept Inventory Questions for Binary Search Trees and Hash Tables	Karpierz u. Wolfman (2014)

Tabelle A.3.: Publikationen zu Fehlvorstellungen zum Thema Datenstrukturen

A.1.4. Hardware

Titel	Quelle
An investigation of children's conceptualisation of computers and how they work	Hammond u. Rogers (2006)
Examining the effects of an instructional intervention on destabilizing learners' misconceptions about the central processing unit	Ioannou u. Angeli (2014)
Inside the Computer: Visualization and Mental Models	Yehezkel u. a. (2001)
Evaluating student understanding of core concepts in computer architecture	Porter u. a. (2013)

Tabelle A.4.: Publikationen zu Fehlvorstellungen zum Thema Hardware

A.1.5. Internet

Titel	Quelle
An investigation of secondary school students' conceptions on how the internet works	Diethelm u. a. (2012)
"Draw me the Web". Impact of mental model of the Web on information search performance of young users	Dinet u. Kitajima (2011)
Wie funktioniert eigentlich das Internet? - Empirische Untersuchung von Schülervorstellungen	Diethelm u. Zumbrägel (2010)

Tabelle A.5.: Publikationen zu Fehlvorstellungen zum Thema Internet

A.1.6. Logik

Titel	Quelle
Describing the What and Why of Students' Difficulties in Boolean Logic	Herman u. a. (2012)
The development of a digital logic concept inventory	Herman (2011)
Proof by Incomplete Enumeration and Other Logical Misconceptions	Herman u. a. (2008)
Limitations in the understanding of mathematical logic by novice computer science students	Almstrum (1994)
Creating the Digital Logic Concept Inventory	Herman u. a. (2010)
Commonsense Computing (episode 6): Logic is Harder than Pie	Vandegrift u. a. (2010)

Tabelle A.6.: Publikationen zu Fehlvorstellungen zum Thema Logik

A.1.7. Nebenläufigkeit

Titel	Quelle
Commonsense Computing (episode 3): Concurrency and Concert Tickets	Lewandowski u. a. (2007)
Gardeners and Cinema Tickets: High School Students' Preconceptions of Concurrency	Kolikant (2001)

Tabelle A.7.: Publikationen zu Fehlvorstellungen zum Thema Nebenläufigkeit

A.1.8. Objektorientierung

Titel	Quelle
Misunderstandings about object-oriented design: experiences using code reviews	Turner u. a. (2008)
Students' understandings of storing objects	Sorva (2007)
Checklists for Grading Object-Oriented CS1 Programs: Concepts and Misconceptions	Sanders u. Thomas (2007)
Identifying novice difficulties in object oriented design	Thomasson u. a. (2006)
Novice Java programmers' conceptions of object and class and variation theory	Eckerdal u. Thuné (2005)
Avoiding object misconceptions	Holland u. a. (1997)

Tabelle A.8.: Publikationen zu Fehlvorstellungen zum Thema Objektorientierung

A.1.9. Programmierung allgemein

Titel	Quelle
Novice Users' Misconceptions of BASIC Programming	Bayman u. Mayer (1982)
Meaningful Categorisation of Novice Programmer Errors	McCall u. Kölling (2014)
Activating "black boxes" instead of opening "zipper" - a method of teaching novices basic CS concepts	Haberman u. Kolikant (2001)
Misconceptions and Attitudes that Interfere with Learning to Program	Clancy (2004)
Tracing Quiz Set to Identify Novices' Programming Misconceptions	Sekiya u. Yamaguchi (2013)
Pascal and High-School Students: A Study of Misconceptions.	Sleeman (1984)
An empirical study of novice program comprehension in the imperative and object-oriented styles	Ramalingam u. Wiedenbeck (1997)
Learning to program-difficulties and solutions	Gomes u. Mendes (2007)
Identifying student misconceptions of programming	Hristova u. a. (2003)
Notional machines and introductory programming education	Sorva (2013)
Improving the mental models held by novice programmers using cognitive conflict and Jeliot visualisations	Ma u. a. (2009)
Study on Difficulties and Misconceptions With Modern Type Systems	Tirronen (2014)
Identifying Student Misconceptions of Programming	Kaczmarczyk u. a. (2010)

Titel	Quelle
Investigating the viability of mental models held by novice programmers	Ma u. a. (2007)
A closer look at tracing explaining and code writing skills in the novice programmer	Venables u. a. (2009)
A study of the difficulties of novice programmers	Lahtinen u. a. (2005)
A diagnosis of beginning programmers' misconceptions of BASIC programming statements	Bayman u. Mayer, Richard (1983)
Alternatives to construct-based programming misconceptions	Spohrer u. Soloway (1986a)
Using SOLO taxonomy to explore students' mental models of the programming variable and the assignment statement	Jimoyiannis (2011)
Student mistakes in an introductory programming course	Sarkar u. a. (2013)
Programming in Java: student-constructed rules	Fleury (2000)
Developing and validating test items for first-year computer science courses	Vahrenhold u. Paul (2014)
Assignment and sequence: why some students can't recognise a simple swap	Simon (2011)
Novice programming errors: misconceptions or misrepresentations?	Christiaen (1988)
Two misconceptions about structured Programming	Denning (1975)
Exploring Programming Misconceptions	Sirkiä u. Sorva (2012)

Tabelle A.9.: Publikationen zu Fehlvorstellungen zum Thema Programmierung

A.1.10. Rekursion

Titel	Quelle
What do novice programmers know about recursion	Kahney (1983)
Conceptual models and cognitive learning styles in teaching recursion	Wu u. a. (1998)
The Case of Base Cases: Why are They so difficult to Recognize? Student Difficulties with Recursion	Haberman u. Averbuch (2002)
Mental Models of Recursion	Dicheva u. Close (2003)

Tabelle A.10.: Publikationen zu Fehlvorstellungen zum Thema Rekursion

A.1.11. Variablen

Titel	Quelle
The same but different - Students' understandings of primitive and object variables	Sorva (2008)
Understanding the programming variable concept with animated interactive analogies	Doukakis u. a. (2007)

Tabelle A.11.: Publikationen zu Fehlvorstellungen zum Thema Variablen

A.1.12. Sonstiges

Titel	Quelle
Typische Schülerfehler bei Informatikaufgaben	Hansky (2010)
Some common misconceptions about performance modeling and validation	Cao (1993)
The FCS1 : A Language Independent Assessment of CS1 Knowledge	Tew u. Guzdial (2011)
Analyzing the Strength of Undergraduate Misconceptions About Software Engineering Categories and Subject Descriptors	Sudol u. Jaspan (2010)
Constructivism in computer science education	Ben-Ari (2001)
Common sense computing (episode 4): debugging	Simon u. a. (2008)
Putting threshold concepts into context in computer science education	Eckerdal u. a. (2006)
Impact of alternative introductory courses on programming concept understanding	Tew u. a. (2005)
Assessing Fundamental Introductory Computing Concept Knowledge	Tew (2010)
Concept inventories in computer science for the topic discrete mathematics	Almstrum u. a. (2006)

Tabelle A.12.: Publikationen zu Fehlvorstellungen zu sonstigen Themen

A.2. Analogien und Metaphern

Titel	Quelle
Difficulties in Learning Inheritance and Polymorphism	Liberman u. a. (2011)
Metaphors and Analogies for Teaching Algorithms	Forišek u. Steinová (2012)
Understanding the programming variable concept with animated interactive analogies	Doukakis u. a. (2007)

Tabelle A.13.: Publikationen zu Fehlvorstellungen zum Thema Analogien und Metaphern

A.3. Verwandte Fächer

Titel	Quelle
Chemiedidaktik. Diagnose und Korrektur von Schülervorstellungen	Barke (2006)

Tabelle A.14.: Publikationen zu Fehlvorstellungen aus verwandten Fächern

A.4. Sonstige Publikationen

Titel	Quelle
Presenting Computer Science Concepts to High School Students	Bell u. a. (2014)
Guide to Teaching Computer Science - An Activity-Based Approach	Hazzan u. a. (2011)

Tabelle A.15.: Sonstige Publikationen

Anhang B.

Rohdaten der Erhebungen

Die Rohdaten zur Erhebung sind über das „Open Science Framework“ verfügbar.

Erhebung von zentralen Konzepten und der Häufigkeit von Fehlvorstellungen:
URL: `https://osf.io/abh82`
DOI: 10.17605/OSF.IO/ABH82

Hauptstudie Wissen über Fehlvorstellungen:
URL: `https://osf.io/kq4bw`
DOI: 10.17605/OSF.IO/KQ4BW

Anhang C.

Testinstrumente

C.0.1. Erhebung von zentralen Konzepten und der Häufigkeit von Fehlvorstellungen

Anschreiben

Hallo {FIRSTNAME}{LASTNAME},

ich kontaktiere Sie, da Sie im aktuellen oder im vergangenen Semester eine Lehrveranstaltung für Studierende im Bachelorstudiengang Informatik angeboten haben und möchte Sie freundlich bitten, an einer Umfrage teilzunehmen.

Ziel der Umfrage ist es, zentrale Konzepte der Informatik hinsichtlich ihrer Relevanz in der Lehre und Fehlvorstellungen von Lernenden zu bewerten. Die Ergebnisse werden anschließend im Rahmen eines Forschungsprojektes dazu genutzt, ein Testinstrument zur Messung des Wissens von Lehrenden über Fehlvorstellungen und Lernschwierigkeiten zu erstellen. Um an dieser Umfrage teilzunehmen, klicken Sie bitte auf den folgenden Link: {SURVEYURL}

Mit freundlichen Grüßen
Laura Ohrndorf

Messinstrument

Hinweis: Das auf den folgenden Seiten abgebildete Messinstrument wurde nur online in Limesurvey eingesetzt und wird hier als unbearbeiteter Export der Online-Umfrage vorgestellt. Die Formatierung dieser Version unterscheidet sich in einigen Details von der im Webbrowser dargestellten Version.

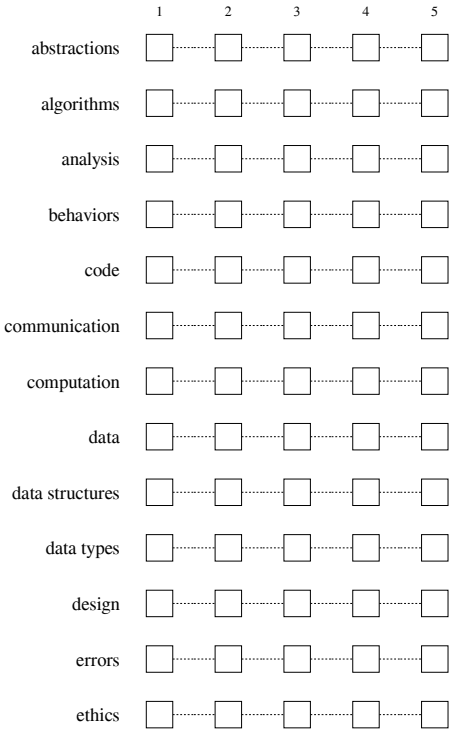
Willkommen zur Umfrage zu Konzepten der Informatik!

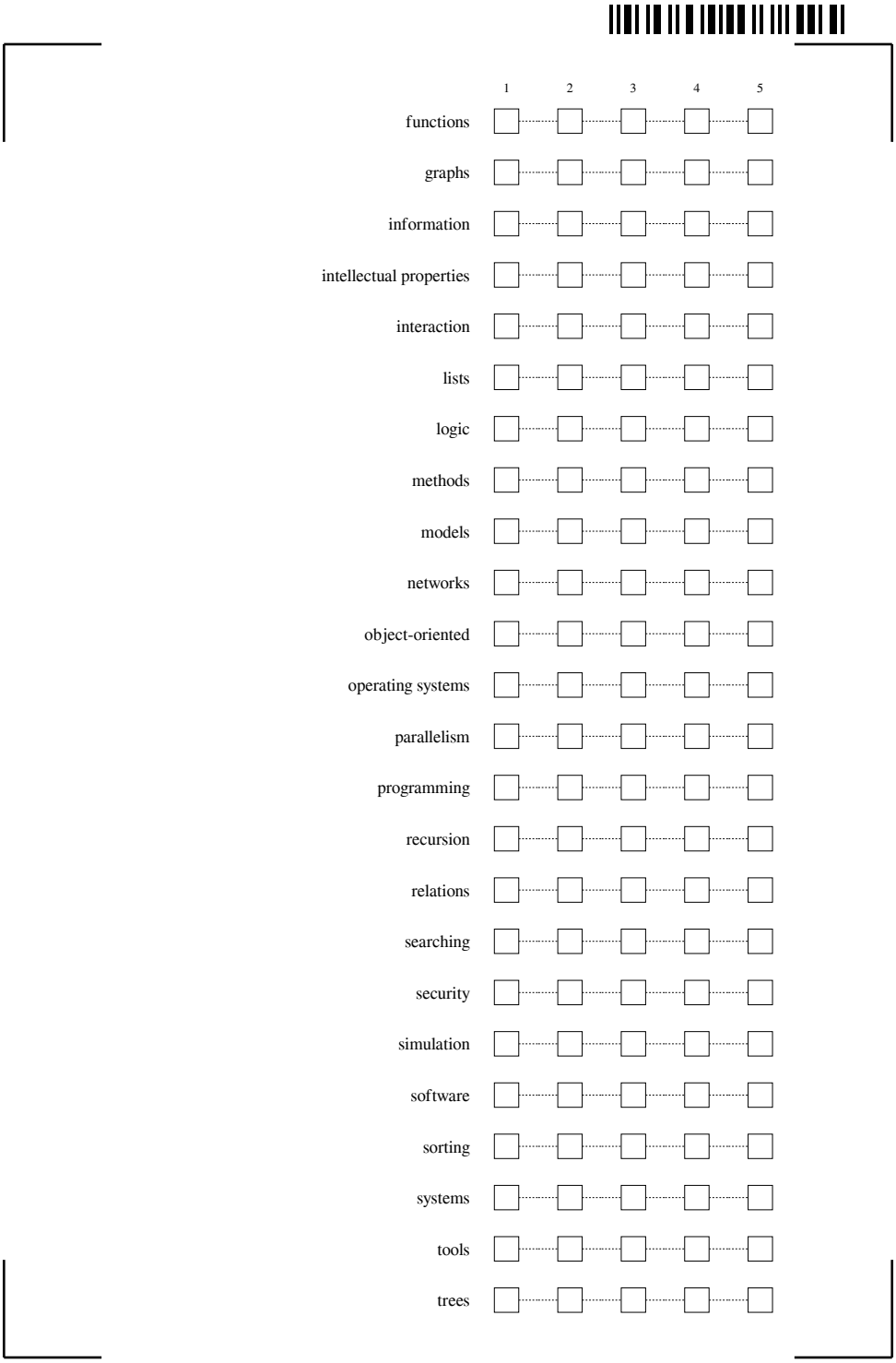
Ziel der Umfrage ist es, zentrale Konzepte der Informatik hinsichtlich ihrer Relevanz in der Lehre und ihrer Schwierigkeit für die Lernenden zu bewerten. Die Ergebnisse werden anschließend im Rahmen eines Forschungsprojektes dazu genutzt, ein Testinstrument zur Messung des Wissens von Lehrenden über Fehlvorstellungen und Lernschwierigkeiten zu erstellen.

Ich würde mich sehr freuen, wenn Sie sich die Zeit nehmen würden, die Umfrage auszufüllen. Der Zeitbedarf beträgt ca. 8-10 Minuten.

Teil A: Fundamentale Ideen - Horizontalkriterium

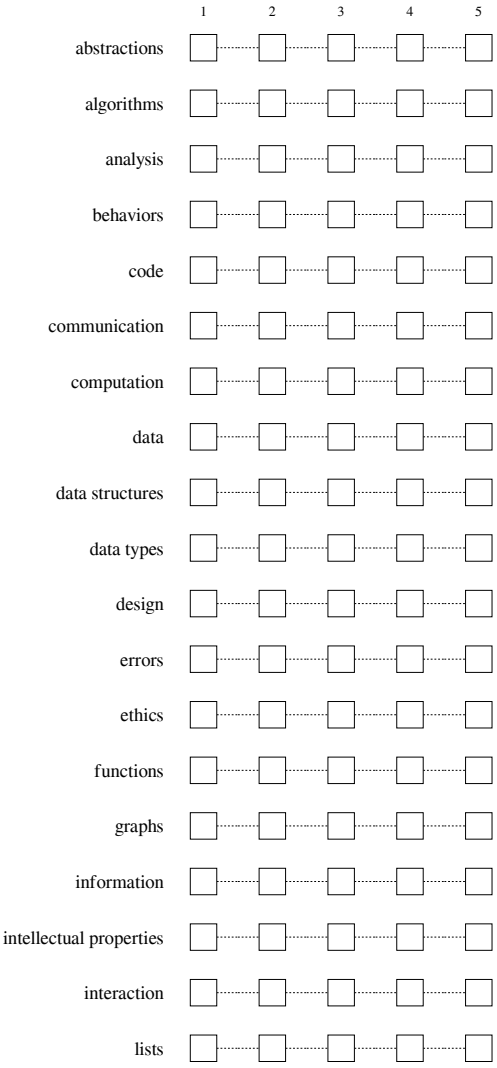
A1. **Bitte beurteilen Sie, inwiefern Sie der folgenden Aussage zustimmen: "*Das Konzept ist in verschiedenen Bereichen der Informatik anwendbar oder erkennbar.*" (von 1 = "ich stimme zu" bis 5 = "ich stimme nicht zu")**

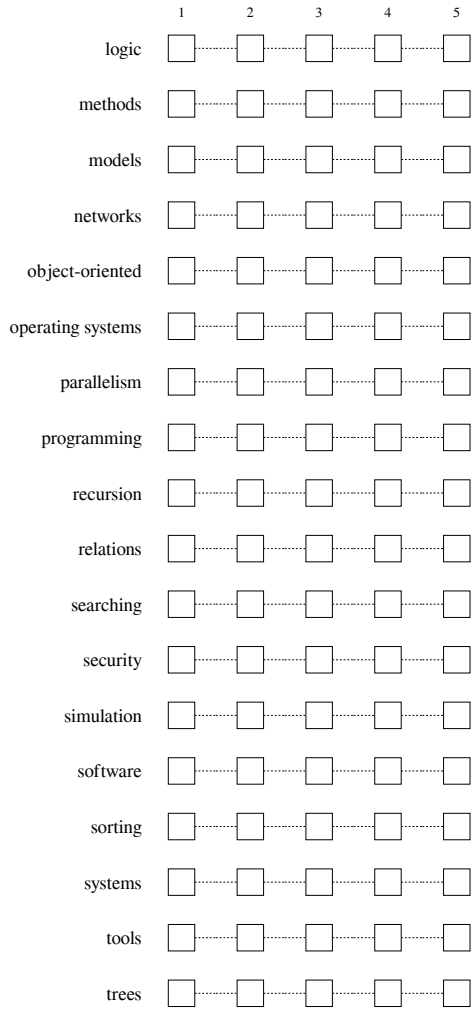



1
2
3
4
5
functions
graphs
information
intellectual properties
interaction
lists
logic
methods
models
networks
object-oriented
operating systems
parallelism
programming
recursion
relations
searching
security
simulation
software
sorting
systems
tools
trees

Teil B: Fundamentale Ideen - Vertikalkriterium

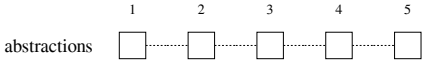
B1. **Bitte beurteilen Sie, inwiefern Sie der folgenden Aussage zustimmen: "*Das Konzept kann auf jedem intellektuellen Niveau aufgezeigt und vermittelt werden.*" (von 1 = "ich stimme zu" bis 5 = "ich stimme nicht zu") Hinweis: Hier geht es um das Konzepte als solches, nicht die konkreten Inhalte. Informatische Themen lassen sich z.B. auch mit einer visuellen Programmiersprache verdeutlichen und sind daher schon im Kindesalter vermittelbar.**

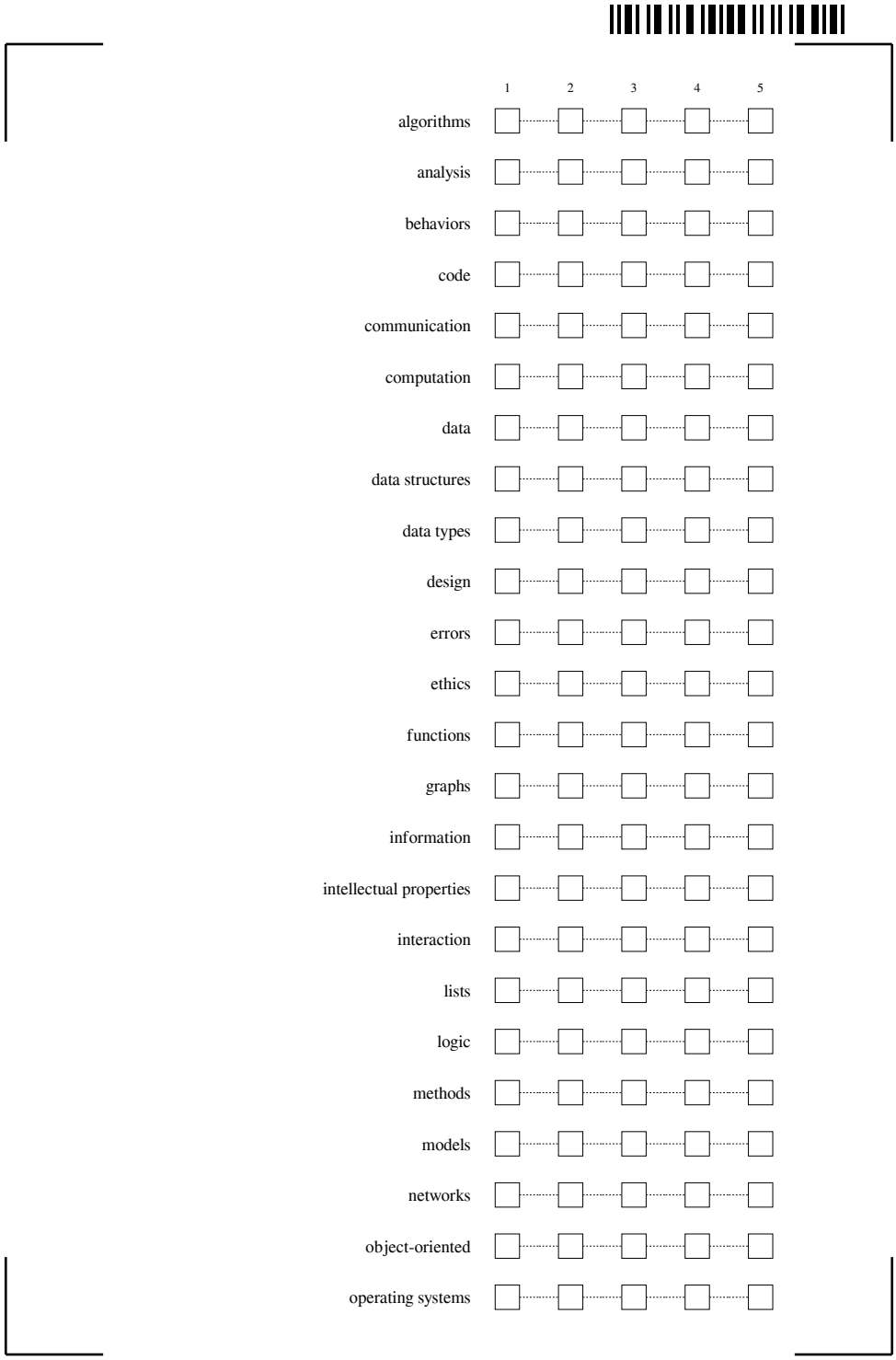


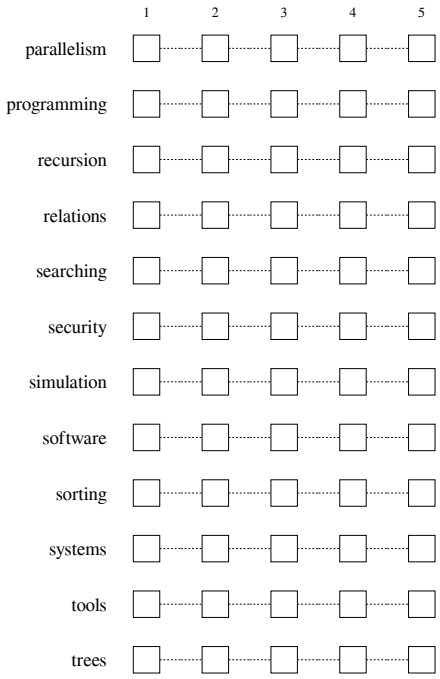


Teil C: Fundamentale Ideen - Zeitkriterium

C1. Bitte beurteilen Sie, inwiefern Sie der folgenden Aussage zustimmen: "*Das Konzept lässt sich in der Geschichte der Informatik deutlich wahrnehmen und wird auch längerfristig relevant bleiben.*" (von 1 = "ich stimme zu" bis 5 = "ich stimme nicht zu").

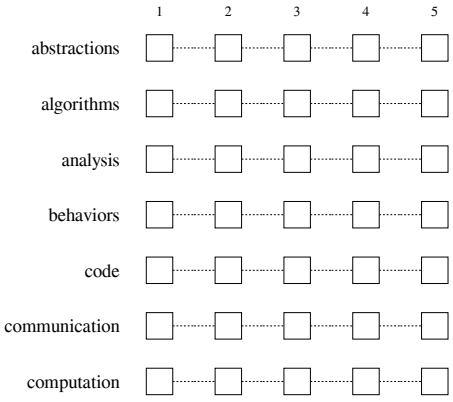


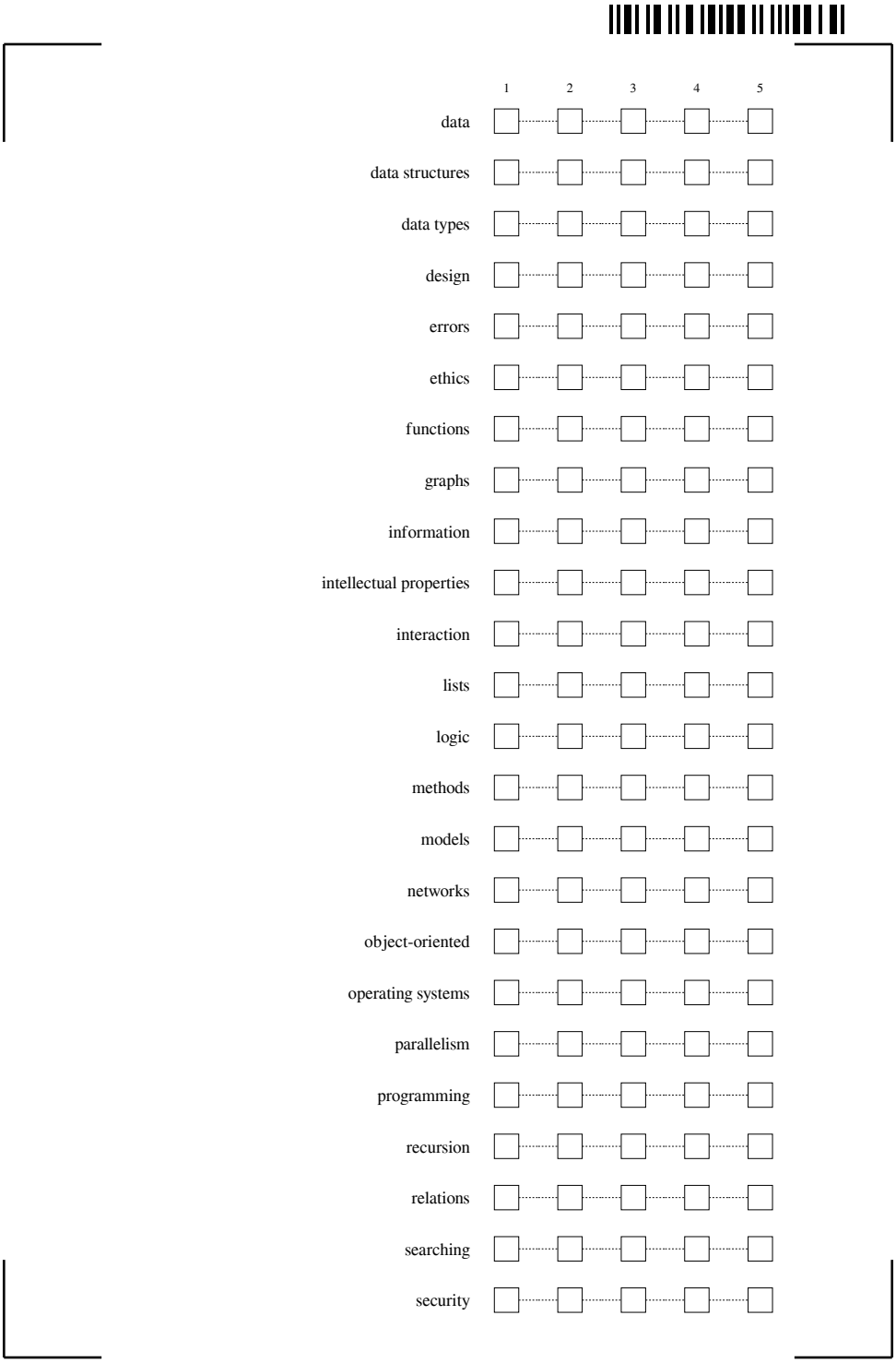

1
2
3
4
5
algorithms
analysis
behaviors
code
communication
computation
data
data structures
data types
design
errors
ethics
functions
graphs
information
intellectual properties
interaction
lists
logic
methods
models
networks
object-oriented
operating systems

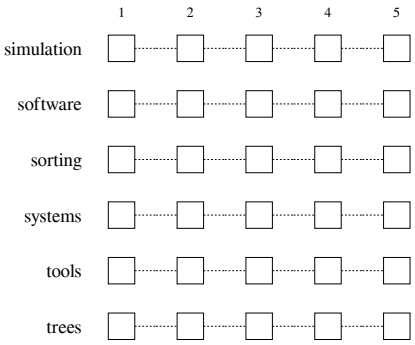


Teil D: Fundamentale Ideen - Sinnkriterium

D1. **Bitte beurteilen Sie, inwiefern Sie der folgenden Aussage zustimmen: "*Das Konzept hat einen lebensweltlichen Bezug und ist auch im Alltag nachweisbar.*" (von 1 = "ich stimme zu" bis 5 = "ich stimme nicht zu").**




1
2
3
4
5
data
data structures
data types
design
errors
ethics
functions
graphs
information
intellectual properties
interaction
lists
logic
methods
models
networks
object-oriented
operating systems
parallelism
programming
recursion
relations
searching
security

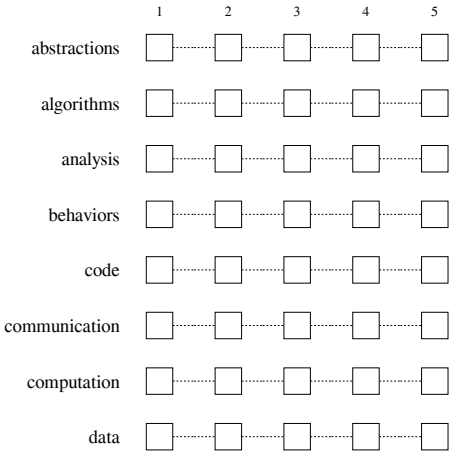


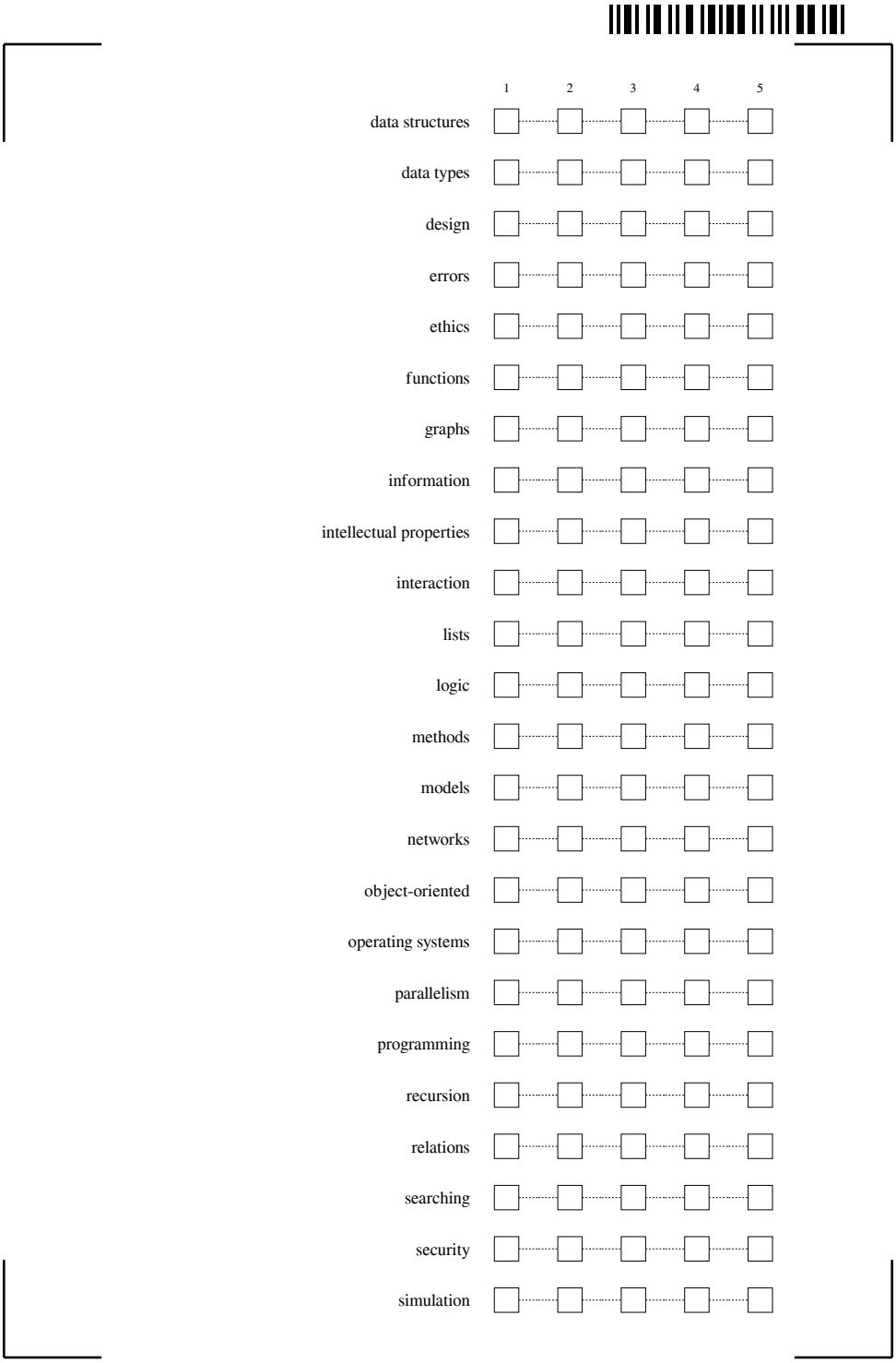
Teil E: Fehlvorstellungen

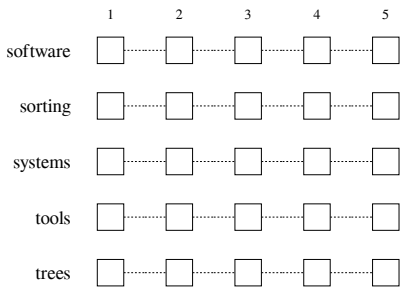
E1. **Bitte bewerten Sie die Häufigkeit von Fehlvorstellungen, die bei Lernenden zu diesem Konzept auftreten auf einer Skala von 1 (Fehlvorstellungen treten selten bis nie auf) bis 5 (Fehlvorstellungen treten sehr häufig auf)). Betrachten Sie dabei das Konzept selbst, nicht konkrete Inhalte.**

Wichtig: Bei einer Fehlvorstellung handelt es sich um falsches oder unvollständiges Wissen, welches nicht dem fachlich korrekten Wissen entspricht. Dieses Kriterium bezieht sich somit nicht auf die Schwierigkeit von Konzepten oder die Häufigkeit von Fehlern. Fehlvorstellungen können auch außerhalb von Schule und Universität erworben werden und erst später identifiziert werden.

Hinweis: bitte bewerten Sie die Konzepte unabhängig voneinander, nicht in Relation zueinander.




1 2 3 4 5
data structures
data types
design
errors
ethics
functions
graphs
information
intellectual properties
interaction
lists
logic
methods
models
networks
object-oriented
operating systems
parallelism
programming
recursion
relations
searching
security
simulation



Teil F: Daten

F1. Wo liegt ihr vornehmliches Tätigkeitsfeld?

- ☐ Universität
- ☐ Schule
- ☐ Sonstiges

F2. In welchem Gebiet liegt der inhaltliche Schwerpunkt ihrer Lehrtätigkeit?

- ☐ Fachwissenschaft Informatik
- ☐ Fachdidaktik Informatik
- ☐ Sonstiges

F3. Seit wie vielen Jahren sind Sie in der Lehre tätig?

- ☐ 0-5
- ☐ 6-10
- ☐ 11-15
- ☐ >15

Vielen Dank, dass Sie an der Umfrage teilgenommen haben!

Bei Fragen oder Anmerkungen wenden Sie sich bitte an: laura.ohrndorf@uni-siegen.de

C.0.2. Hauptstudie

Anschreiben

Sehr geehrte Damen und Herren,

wir möchten Sie herzlich zur Teilnahme an unserer Befragung zum Thema Fehlvorstellungen in der Informatik einladen, die wir im Rahmen eines Forschungsprojektes an der Universität Paderborn durchführen.

Mit dieser möchten wir erheben, wie Sie Fehlvorstellungen von Lernenden einschätzen und interpretieren. Wir betreten damit ein recht unerforschtes Gebiet in der Informatik und würden uns freuen, wenn Sie uns dabei unterstützen könnten.

Die Bearbeitung dauert etwa 20 Minuten. Wenn Sie mit der Thematik einer Frage nicht vertraut sind, können Sie diese natürlich auslassen und zur nächsten Frage weitergehen.

Ihre Antworten werden anonymisiert erhoben. Das Ausfüllen des Fragebogens lässt keine Rückschlüsse auf Ihre Identität zu.

Um an dieser Umfrage teilzunehmen, klicken Sie bitte auf den unten stehenden Link: {SURVEYURL}

Gerne können Sie den Link auch an interessierte Kollegen oder Kommilitonen weitergeben.

Vielen Dank und mit freundlichen Grüßen,
Laura Ohrndorf (laura.ohrndorf@upb.de)

Messinstrument

Hinweis: Das auf den folgenden Seiten abgebildete Messinstrument wurde nur online in Limesurvey eingesetzt und wird hier als unbearbeiteter Export der Online-Umfrage vorgestellt. Die Formatierung dieser Version unterscheidet sich in einigen Details von der im Webbrowser dargestellten Version.

Herzlich Willkommen und vielen Dank dafür, dass Sie sich für diese Umfrage interessieren! Diese Befragung ist Teil eines Forschungsvorhabens zur Messung des Wissens über Fehlvorstellungen in der Informatik, welches an der Universität Paderborn durchgeführt wird. Bei einer Fehlvorstellung handelt es sich um falsches oder unvollständiges Wissen, welches nicht dem fachlich korrekten Wissen entspricht. Häufig wird dieses außerhalb von Schule und Universität erworben. Es geht hier somit nicht um Flüchtigkeitsfehler wie z.B. Syntaxfehler, sondern um das grundlegende Verständnis. Auf viele der hier gestellten Fragen gibt es keine eindeutige Antwort. Wenn Sie also unsicher sind oder ihre Antwort für ungewöhnlich halten, geben Sie diese bitte trotzdem an. Wenn Sie Fragen oder Anmerkungen zur Befragung haben, wenden Sie sich bitte an: laura.ohrndorf@uni-paderborn.de
Vielen Dank für Ihre Unterstützung!

Teil A: Demographische Daten

An dieser Stelle möchten wir Sie zunächst bitten, einige demographische Fragen zu beantworten. Diese Fragen helfen uns, die Ergebnisse der Umfrage auszuwerten. Die Daten werden selbstverständlich nicht für Ihre Person ausgewertet, sondern für Gruppen, denen Sie anhand ihrer Antwort zugeordnet werden.

A1. Zu welcher Altersgruppe gehören Sie?

- ☐ Unter 20
- ☐ 20-29
- ☐ 30-39
- ☐ 40-49
- ☐ 50-59
- ☐ 60 und älter

A2. Welchen Beruf haben Sie?

Falls mehrere Antworten zutreffen, wählen Sie bitte diejenige aus, die dem Großteil ihrer Arbeitszeit entspricht.

- ☐ Student
- ☐ Professor oder wissenschaftlicher Mitarbeiter an einer Universität
- ☐ Lehrer Informatik
- ☐ sonstige Berufstätigkeit im Bereich Informatik (z.B.Programmierer oder Systemadministrator)
- ☐ Sonstiges

A3. Welches Fach studieren Sie?

- ☐ Informatik
- ☐ Lehramt Informatik
- ☐ Sonstiges

A4. Sind Sie im Forschungsgebiet Fachdidaktik Informatik aktiv?

Aktiv = mindestens eine wissenschaftliche Veröffentlichung in der Fachdidaktik Informatik

- ☐ Ja
- ☐ Nein

A5. Haben Sie im Fach Informatik Lehrveranstaltungen gehalten bzw. Unterricht durchgeführt?

Diese Frage bezieht sich sowohl auf Lehrveranstaltungen an Universitäten, als auch auf Informatikunterricht an Schulen.

- ☐ Ja
- ☐ Nein

A6. Welche Gruppen haben Sie unterrichtet?

- ☐ Schüler Sekundarstufe 1
- ☐ Schüler Sekundarstufe 2
- ☐ Studierende (Universität)
- ☐ Sonstige

A7. Wie lange haben Sie in der Informatik unterrichtet?

- ☐ weniger als 1 Jahr
- ☐ 1-5 Jahre
- ☐ 6-10 Jahre
- ☐ mehr als 10 Jahre

A8. Haben Sie eine didaktische Ausbildung?

- ☐ Abgeschlossenes Studium Lehramt Informatik
- ☐ Abgeschlossenes Studium Lehramt Sonstige
- ☐ Sonstiges (z.B. didaktische Fortbildung oder Weiterbildung)
- ☐ keine

Teil B: Algorithmen

Vergleichen Sie bitte die folgenden beiden Algorithmen, die das Produkt zweier Zahlen durch Addition berechnen.

Algorithmus 1 Algorithmus 2 mult = 0 x = input("Enter number 1: ") y = input("Enter number 2: ") for i in range(1,x+1): mult = mult + y print(mult) mult = 0 x = input("Enter number 1: ") y = input("Enter number 2: ") if x

B1. **Welcher Algorithmus ist effizienter?**

Algorithmus 1 ☐

Algorithmus 2 ☐

Keiner (Effizienz ist gleich) ☐

B2. **Nennen Sie mindestens eine Fehlvorstellung, die zu einer falschen Beantwortung dieser Frage führen könnte:**

Teil C: Sortieralgorithmen

C1. **Ein Student soll einen Algorithmus beschreiben, der die folgenden zehn Zahlen aufsteigend nach ihrer Größe sortiert: 20, 341, 4, 38, 93, 120, 92, 250, 171, 48 Student: *„Ich teile die Zahlen erst einmal nach ihrer Größe auf, also einmal die mit einer Ziffer, dann die mit zwei Ziffern, dann die mit drei. In der ersten Gruppe ist nur eine Zahl, also ist das schon einmal die kleinste. Dann teile ich die zweite Gruppe in weitere Gruppen auf, indem ich die erste Ziffer anschaue und vergleiche. So gehe ich dann weiter vor, bis alle Zahlen angeordnet sind.“* Ist dieses Vorgehen richtig? Welche Fehlvorstellung könnte hier vorliegen?**

Teil D: Logik

Für die Zubereitung eines Auflaufs gelten die folgenden Regeln:

Wenn der Auflauf Nudeln enthält, muss er auch Käse enthalten.

Es ist nicht möglich, dass der Auflauf entweder nur Brokkoli oder nur Schinken enthält.

Der Auflauf enthält nicht gleichzeitig Kartoffeln und Hackfleisch.

Entsprechen die folgenden Zutatenlisten den Regeln? Falls nicht, gegen welche Regel verstossen sie?

D1. Zutatenliste 1: Nudeln, Schinken, Brokkoli

- ☐ Richtig
- ☐ Regel 1
- ☐ Regel 2
- ☐ Regel 3

D2. Zutatenliste 2: Nudeln, Hacklfeisch, Brokkoli, Käse

- ☐ Richtig
- ☐ Zutaten verletzten Regel 1
- ☐ Zutaten verletzten Regel 2
- ☐ Zutaten verletzten Regel 3

D3. Zutatenliste 3: Kartoffeln, Brokkoli, Schinken, Käse

- ☐ Richtig
- ☐ Zutaten verletzten Regel 1
- ☐ Zutaten verletzten Regel 2
- ☐ Zutaten verletzten Regel 3

Teil E: Programmierung

E1. **Ein Schüler schreibt den folgenden Programmcode, mit dem zwei eingegebene Zahlen addiert werden sollen.**
x = input("Enter number 1: ") y = input("Enter number 2: ") x + y = sumWelche Fehlvorstellung könnte hier vorhanden sein?

Achtung: Es geht in dieser Aufgabe nicht darum, mögliche Fehler (z.B. Strings als Eingabe) abzufangen.

E2. **Der folgende Programmcode soll ermitteln, ob eine Zahl gerade oder ungerade ist und das Ergebnis ausgeben. Der Programmierer wundert sich, warum keine Ausgabe erfolgt.**

int zahl = 6; if (zahl%2 == 0) { x = "gerade" } else { y = "ungerade" print(x) print(y) } Welche Fehlvorstellung könnte hier vorhanden sein?

Teil F: Datenstrukturen

F1. **Wählen Sie die Analogie aus, die am besten eine Warteschlange in der Informatik (*queue*) repräsentiert und begründen Sie ihre Antwort:**

- ☐ Warteschlange an einer Supermarktkasse
- ☐ Vergabe von aufsteigenden Nummern beim Betreten eines Wartezimmers, die einzeln aufgerufen werden
- ☐ Auslieferung der Bestellungen in einem Restaurant

F2. **Begründen Sie ihre Entscheidung**

Teil G: Analogien

Analogien werden in der Informatik häufig verwendet, um unbekannte Konzepte zu erklären und den Lernprozess zu fördern.

Der Arbeitsspeicher eines Computers kann als Eis (im Sinne von gefrorenem Wasser) beschrieben werden. Es hat, ebenso wie Speicher, zwei Zustände: flüssig und gefroren. Um zwischen diesen zu wechseln, braucht man Energie. Wenn Eis schmilzt, geht der vorherige Zustand (d.h. die Speicherbelegung) verloren.

G1. **Nennen Sie mindestens eine Eigenschaft von Arbeitsspeicher, für die sich diese Analogie nicht eignet und daher zu einer Fehlvorstellung bei Lernenden führen kann. Begründen Sie ihre Antwort kurz.**

Teil H: Array

H1.

Die Lernenden sollen den folgenden Programmcode nachvollziehen und den Wert von array1 angeben:

array1 = [6, 9, 4, 2, 8]
array2 = [5, 2, 2, 1]
array1[2] = array2[3]
array1[4] = array2[1]array1[3] = array2[2] + 2

Welchen Wert hat die Variable array1 nun?

- [6, 9, 1, 2, 9] ☐
- [6, 9, 1, 4, 2] ☐
- [6, 2, 4, 2, 9] ☐
- [2, 6, 4, 1, 4] ☐

H2. **Die Antwort eines Lernenden lautet:**

array1 = [6, 2, 4, 5, 8]

Welche Fehlvorstellung könnte hier vorliegen?

Sie haben das Ende der Umfrage erreicht. Vielen Dank für Ihre Unterstützung!

Wenn Sie Fragen oder Anmerkungen zur Befragung haben, wenden Sie sich bitte an: laura.ohrndorf@uni-paderborn.de

Literaturverzeichnis

[ACM/IEEE-CS Joint Task Force on Computing Curricula 2013] ACM/IEEE-CS Joint Task Force on Computing Curricula: Computer Science Curricula 2013 / ACM Press and IEEE Computer Society Press. Version: 2013. http://dx.doi.org/10.1145/2534860. 2013. – Forschungsbericht

[Adams u. Wieman 2011] Adams, Wendy K. ; Wieman, Carl E.: Development and Validation of Instruments to Measure Learning of Expert-Like Thinking. In: *International Journal of Science Education* 33 (2011), Nr. 9, S. 1289–1312

[Almstrum 1994] Almstrum, Vicki L.: *Limitations in the understanding of mathematical logic by novice computer science students*, University of Texas, Austin, Diss., 1994. – 201 S.

[Almstrum u. a. 2006] Almstrum, Vicki L. ; Henderson, Peter B. ; Harvey, Valerie ; Heeren, Cinda ; Marion, William ; Riedesel, Charles ; Soh, Leen-Kiat ; Tew, Allison E.: *Concept inventories in computer science for the topic discrete mathematics*. New York, New York, USA, 2006

[An 2004] An, Shuhua: The Pedagogical Content Knowledge of Middle School, Mathematics Teachers in China and the U.S. In: *Journal of Mathematics Teacher Education* 7 (2004), Nr. 2, S. 145–172

[Anderson u. Krathwohl 2001] Anderson, Lorin W. ; Krathwohl, David R.: *A Taxonomy for Learning, Teaching, and Assessing: a Revision of Bloom's Taxonomy of Educational Objectives*. New York : Longman, 2001. – 352 S.

[Bangert-Drowns u. a. 1991] Bangert-Drowns, Robert L. ; Kulik, Chen-Lin C. ; Kulik, James A. ; Morgan, MaryTeresa: The Instructional Effect of Feedback in Test-Like Events. In: *Review of Educational Research* 61 (1991), Nr. 2, S. 213–238

[Barke 2006] Barke, Hans-Dieter: *Chemiedidaktik. Diagnose und Korrektur von Schülervorstellungen*. Heidelberg : Springer, 2006. – 324 S.

[Baumert u. Kunter 2006] Baumert, Jürgen ; Kunter, Mareike: Stichwort : Professionelle Kompetenz von Lehrkräften. In: *Zeitschrift für Erziehungswissenschaft* 9 (2006), Nr. 4, S. 469–520

[Bayman u. Mayer 1982] Bayman, Piraye ; Mayer, Richard E.: Novice Users' Misconceptions of BASIC Programming / California University. Santa Barbara, 1982. – Forschungsbericht

[Bayman u. Mayer, Richard 1983] Bayman, Piraye ; Mayer, Richard, E.: A diagnosis of beginning programmers' misconceptions of BASIC programming statements. In: *Communications of the ACM* 26 (1983), Nr. 9, S. 677–679

[Bell u. a. 2014] Bell, Tim ; Duncan, Caitlin ; Jarman, Sam ; Newton, Heidi: Presenting Computer Science Concepts to High School Students. In: *International Olympiad in Informatics* 8 (2014), 3–19. http://www.mii.lt/olympiads{_}in{_}informatics/files/volume8.pdf{#}page=5{%}5Cnhttp://www.mclibre.org/descargar/docs/revista-oii/oii-08{_}201407.pdf{#}page=5

[Bell u. a. 1996] Bell, Tim ; Fellows, Michael R. ; Witten, Ian: *Computer Science Unplugged*. Computer Science Unplugged, 1996

[Ben-Ari 2001] Ben-Ari, Mordechai: Constructivism in computer science education. In: *Journal of Computers in Mathematics and Science Teaching* 20 (2001), Nr. 1, S. 45–73

[Ben-Ari 2005] Ben-Ari, Mordechai: The concorde doesn't fly anymore. In: *ACM SIGCSE Bulletin* 37 (2005), Nr. 1, S. 196–196

[Ben-Ari u. Kolikant 1999] Ben-Ari, Mordechai ; Kolikant, Yifat Ben-David: Thinking parallel. In: *ACM SIGCSE Bulletin* 31 (1999), S. 13–16

[Bennedsen u. Caspersen 2007] Bennedsen, Jens ; Caspersen, Michael E.: Failure rates in introductory programming. In: *ACM SIGCSE Bulletin* 39 (2007), Nr. 2, S. 32

[Bennedsen u. Caspersen 2008] Bennedsen, Jens ; Caspersen, Michael E.: Exposing the programming process. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4821 LNCS (2008), S. 6–16

[Biggs u. Collis 1982] Biggs, J B. ; Collis, K F.: *Evaluating the quality of learning: the SOLO taxonomy (structure of the observed learning outcome).* Academic Press, 1982 (Educational psychology)

[Biggs 1979] Biggs, John: Individual differences in study processes and the Quality of Learning Outcomes. In: *Higher Education* 8 (1979), Nr. 4, S. 381–394

[Blömeke u. Suhl 2010] Blömeke, Sigrid ; Suhl, Ute: Modellierung von Lehrerkompetenzen. In: *Zeitschrift für Erziehungswissenschaft* 13 (2010), Nr. 3, S. 473–505

[Bonar u. Soloway 1985] Bonar, Jeffrey ; Soloway, Elliot: Preprogramming knowledge: A major source of misconceptions in novice programmers. In: *Human-Computer Interaction* 1 (1985)

[Bortz u. Döring 2006] Bortz, Jürgen ; Döring, Nicola: *Forschungsmethoden und Evaluation: für Human- und Sozialwissenschaftler.* 4. Heidelberg : Springer Medizin Verlag, 2006. – 906 S.

[Boustedt u. a. 2007] Boustedt, Jonas ; Eckerdal, Anna ; McCartney, Robert ; Moström, Jan E. ; Ratcliffe, Mark ; Sanders, Kate ; Zander, Carol: Threshold concepts in computer science. In: *ACM SIGCSE Bulletin* 39 (2007), S. 504

[Bromme 1997] Bromme, Rainer: Kompetenzen, Funktionen und unterrichtliches Handeln des Lehrers. In: *Psychologie des Unterrichts und der Schule* (1997)

[Brown u. Burton 1978] Brown, John S. ; Burton, Richard R.: Diagnostic Models for Procedural Bugs in Basic Mathematical Skills. In: *Journa for Cognitive Science* 2 (1978), Nr. October, S. 155–192

[Brown u. VanLehn 1980] Brown, John S. ; VanLehn, Kurt: Repair theory: A generative theory of bugs in procedural skills. In: *Journal for Cognitive Science* 4 (1980), Nr. 19130, S. 379–426

[Cao 1993] Cao, Xiren: Some common misconceptions about performance modeling and validation. In: *ACM SIGMETRICS Performance Evaluation Review* 21 (1993), Nr. December, S. 11–15. `http://dx.doi.org/10.1145/174215.174217`. – DOI 10.1145/174215.174217. – ISSN 01635999

[Chee 1993] Chee, Yam S.: Applying Gentner's theory of analogy to the teaching of computer programming. In: *International Journal of Man-Machine Studies* 38 (1993), Nr. 3, S. 347–368

[Chen u. a. 2007] Chen, Tzu-Yi ; Lewandowski, Gary ; McCartney, Robert ; Sanders, Kate ; Simon, Beth: Commonsense computing: using student sorting abilities to improve instruction. In: *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*. Covington, Kentucky, USA : ACM, 2007, S. 276–280

[Cheng u. Holyoak 1985] Cheng, P W. ; Holyoak, K J.: Pragmatic reasoning schemas. In: *Journal for Cognitive Psychology* 17 (1985), S. 391–416

[Christiaen 1988] Christiaen, H.: Novice programming errors: misconceptions or misrepresentations? In: *ACM SIGCSE Bulletin* 20 (1988), Nr. 3, S. 5–7

[Clancy 2004] Clancy, Michael: Misconceptions and Attitudes that Interfere with Learning to Program. In: Fincher, Sally (Hrsg.) ; Petri, Marian (Hrsg.): *Computer Science Education Research*. 2004, S. 85–100

[Costa u. Liebmann 1996] Costa, Arthur L. ; Liebmann, Rosemarie M.: *Envisioning Process as Content: Toward a Renaissance Curriculum.* Corwin Press, 1996. – 279 S.

[Danielsiek u. a. 2012] Danielsiek, Holger ; Paul, Wolfgang ; Vahrenhold, Jan: Detecting and Understanding Students' Misconceptions Related to Algorithms and Data Structures. In: *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*. Raleigh, North Carolina, USA : ACM, 2012, S. 21–26

[Denning 1975] Denning, Peter J.: Two misconceptions about structured Programming. In: *Proceedings of the 1975 Annual Conference*. New York, New York, USA : ACM, 1975, S. 214–215

[Denning 2003] Denning, Peter J.: Great principles of computing. In: *Communications of the ACM* 46 (2003), Nr. 11, S. 15

[Denning 2004] Denning, Peter J.: Great Principles in Computing Curricula. In: *Proceedings of the 35th SIGCSE technical symposium on Computer science education - SIGCSE '04* (2004), S. 336–341

[Denning 2005] Denning, Peter J.: Is Computer Science Science? In: *Communications of the ACM* 48 (2005), S. 27–31

[Dicheva u. Close 2003] Dicheva, Darina ; Close, John: Mental Models of Recursion. In: *Journal of Educational Computing Research* 14 (2003), S. 1–23

[Diethelm u. a. 2011] Diethelm, Ira ; Dörge, Christina ; Mesaros, Ana-Maria ; Dünnebier, Malte: Die Didaktische Rekonstruktion für den Informatikunterricht. In: *Informatik in Bildung und Beruf - 14. GI-Fachtagung Informatik und Schule - INFOS 2011*. Münster, Germany, 2011, S. 77–86

[Diethelm u. a. 2012] Diethelm, Ira ; Wilken, Henning ; Zumbrägel, Stefan: An investigation of secondary school students' conceptions on how the internet works. In: *Proceedings of the 12th Koli Calling International Conference on Computing Education Research - Koli Calling '12* (2012), S. 67–73

[Diethelm u. Zumbrägel 2010] Diethelm, Ira ; Zumbrägel, Stefan: Wie funktioniert eigentlich das Internet? - Empirische Untersuchung von Schülervorstellungen. In: *DDI* (2010). `http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Wie+funktioniert+eigentlich+das+Internet+?+-+Empirische+Untersuchung+von+Sch{\protect\unhbox\voidb@x\bgroup\U@D1ex{\setbox\z@\hbox{\char127}\dimen@-.45ex\advance\dimen@\ht\z@}\accent127\fontdimen5\font\U@Du\egroup}l+ervorstellungen{#}0http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Wie+funktioniert+eigentlich+das+In`

[Dinet u. Kitajima 2011] Dinet, Jérôme ; Kitajima, Muneo: "Draw me the Web". Impact of mental model of the Web on information search performance of young users. In: *Ihm* (2011). ISBN 9781450308229

[Doukakis u. a. 2007] Doukakis, Dimitrios ; Grigoriadou, Maria ; Tsaganou, Grammatiki: Understanding the programming variable concept with animated interactive analogies. (2007), S. 1–8

[Eckerdal u. a. 2006] Eckerdal, Anna ; McCartney, Robert ; Moström, Jan E. ; Ratcliffe, Mark ; Sanders, Kate ; Zander, Carol: Putting threshold concepts into context in computer science education. In: *ACM SIGCSE Bulletin* 38 (2006), Nr. 3, S. 103

[Eckerdal u. Thuné 2005] Eckerdal, Anna ; Thuné, M: Novice Java programmers' conceptions of object and class, and variation theory. In: *ACM SIGCSE Bulletin* (2005), S. 89–93

[Ehlert u. Schulte 2009] Ehlert, Albrecht ; Schulte, Carsten: Empirical Comparison of Objects-First and Objects-Later. In: *Proceedings of the Fifth International Workshop on Computing Education Research Workshop*. Berkeley, CA, USA : ACM, 2009, S. 15–26

[Epp 2003] Epp, Susanna S.: The Role of Logic in Teaching Proof. In: *American Mathematical Monthly* 110 (2003), S. 886–899

[Fensham 2002] Fensham, Peter J.: Science Content as Problematic: Issues for Research. In: *Research in Science Education - Past, Present, and Future* (2002), S. 27–41

[Flanagan 1954] Flanagan, John C.: The Critical Incident Technique. In: *Psychological Bulletin* 51 (1954), Nr. 4, S. 327–358

[Fleury 2000] Fleury, Ann E.: Programming in Java: student-constructed rules. In: *ACM SIGCSE Bulletin* 32 (2000), 197–201. `http://dx.doi.org/10.1145/331795.331854`. – DOI 10.1145/331795.331854. – ISBN 1581132131

[Forišek u. Steinová 2012] Forišek, Michal ; Steinová, Monika: Metaphors and Analogies for Teaching Algorithms. In: *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12* (2012), S. 15

[Fuller 1973] Fuller, Frances F.: Teacher education and the psychology of behavior change. In: *Annual Meeting American Psychol. Assoc.*, 1973

[Fuller u. a. 2007] Fuller, Ursula ; Riedesel, Charles ; Thompson, Errol ; Johnson, Colin G. ; Ahoniemi, Tuukka ; Cukierman, Diana ; Hernán-Losada, Isidoro ; Jackova, Jana ; Lahtinen, Essi ; Lewis, Tracy L. ; Thompson, Donna M.: Developing a computer science-specific learning taxonomy. In: *Working group reports on ITiCSE on Innovation and technology in computer science education - ITiCSE-WGR 2007*. New York, New York, USA : ACM Press, 2007, S. 152

[Gal-Ezer u. Zur 2003] Gal-Ezer, Judith ; Zur, Ela: The efficiency of algorithms - Misconceptions. In: *Computers and Education* 42 (2003), S. 215–226

[Goldman u. a. 2008] Goldman, Ken ; Gross, Paul ; Heeren, Cinda ; Herman, Geoffrey L. ; Kaczmarczyk, Lisa ; Loui, Michael C. ; Zilles, Craig: Identifying Important and Difficult Concepts in Introductory Computing Courses using a Delphi Process. In: *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*. Portland, Oregon, USA : ACM, 2008, S. 256–250

[Goldwasser u. Letscher 2008] Goldwasser, Michael H. ; Letscher, David: Teaching an Object-Oriented CS1 - with Python. In: *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*. Madrid, Spain : ACM, 2008, S. 42–46

[Gomes u. Mendes 2007] Gomes, Anabela ; Mendes, António José Nunes: Learning to program-difficulties and solutions. In: *International Conference on Engineering Education* (2007), 1–5. `http://ineer.org/Events/ICEE2007/papers/411.pdf`

[Grimm 2005] Grimm, R: *Digitale Kommunikation*. Oldenbourg, 2005

[Gupta 2007] Gupta, Gopal K.: Computer science curriculum developments in the 1960s. In: *IEEE Annals of the History of Computing* 29 (2007), S. 40–54

[Haberman u. Averbuch 2002] Haberman, Bruria ; Averbuch, Haim: The Case of Base Cases: Why are They so difficult to Recognize? Student Difficulties with Recursion. In: *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education*, ACM Press, 2002, S. 84–88

[Haberman u. Kolikant 2001] Haberman, Bruria ; Kolikant, Yifat Ben-David: Activating "black boxes" instead of opening "zipper" - a method of teaching novices basic CS concepts. In: *ACM SIGCSE Bulletin* 33 (2001), sep, Nr. 3, S. 41–44

[Häder u. Häder 2000] Häder, Michael ; Häder, Sabine: Die Delphi-Methode als Gegenstand methodischer Forschungen. In: *Die Delphi-Technik in den Sozialwissenschaften*. Springer, 2000, S. 11–31

[Hammerer 2001] Hammerer, Franz: Der Fehler – eine pädagogische Schlüsselsituation und Herausforderung. In: *Erziehung und Unterricht* 1-2 (2001), S. 37–50

[Hammond u. Rogers 2006] Hammond, Michael ; Rogers, Philip: An investigation of children's conceptualisation of computers and how they work. In: *Education and Information Technologies* 12 (2006), Nr. 1, S. 3–15

[Hansky 2010] Hansky, Stefanie: *Typische Schülerfehler bei Informatikaufgaben*, Diss., 2010

[Härtig 2014] Härtig, Hendrik: Das Force Concept Inventory: Vergleich einer offenen und einer geschlossenen Version. In: *Physik und Didaktik in Schule und Hochschule* 13 (2014), Nr. 1, S. 53–61

[Hazzan u. a. 2008] Hazzan, Orit ; Gal-Ezer, Judith ; Blum, Lenore: A Model for High School Computer Science Education: The Four Key Elements That Make It! In: *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*. New York, NY, USA : ACM, 2008, S. 281–285

[Hazzan u. a. 2011] Hazzan, Orit ; Lapidot, Tami ; Ragonis, Noa: *Guide to Teaching Computer Science - An Activity-Based Approach*. London : Springer, 2011. – 260 S.

[Heinze 2005] Heinze, Aiso: Mistake-Handling Activities in the Mathematics Classroom. In: *29th Conference of the International Group fot the Psychology of Mathematics Education* Bd. 3, 2005, S. 105–112

[Herman u. a. 2008] Herman, Geoffrey L. ; Kaczmarczyk, Lisa ; Loui, Michael C. ; Zilles, Craig: Proof by Incomplete Enumeration and Other Logical Misconceptions. In: *Proceeding of the fourth international workshop on Computing education research - ICER '08*. Sydney, Australia, 2008, S. 59–70

[Herman u. a. 2012] Herman, Geoffrey L. ; Loui, Michael C. ; Kaczmarczyk, Lisa ; Zilles, Craig: Describing the What and Why of Students' Difficulties in Boolean Logic. In: *ACM Transactions on Computing Education* 12 (2012), Nr. 1, S. 1–28

[Herman u. a. 2010] Herman, Geoffrey L. ; Loui, Michael C. ; Zilles, Craig: Creating the Digital Logic Concept Inventory. In: *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*. Milwaukee, Wisconsin, USA : ACM, 2010, S. 102–106

[Herman 2011] Herman, Geoffrey L.: *The development of a digital logic concept inventory*, University of Illinois at Urbana-Champaign, Diss., 2011. – 294 S.

[Hestenes u. a. 1992] Hestenes, David ; Wells, Malcolm ; Swackhamer, Gregg: Force Concept Inventory. In: *The Physics eacher* 30 (1992), S. 141–158

[Hill u. a. 2008] Hill, Heather C. ; Loewenberg Ball, Deborah ; Schilling, Stephen G.: Unpacking Pedagogical Content Knowledge: Conceptualizing and Measuring Teachers' Topic-Specific Knowledge of Students. In: *Journal for Research in Mathematics Education* 39 (2008), Nr. 4, S. 372–400

[Holland u. a. 1997] Holland, Simon ; Griffiths, Robert ; Woodman, Mark: Avoiding object misconceptions. In: *Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education* (1997), S. 131–134

[Hristova u. a. 2003] Hristova, Maria ; Misra, Ananya ; Rutter, Megan ; Mercuri, Rebecca: Identifying and Correcting Java programming Errors for Introductory Computer Science Students. In: *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, 2003, S. 153–156

[Humbert 2006] Humbert, Ludger: *Didaktik der Informatik, mit praxiserprobtem Unterrichtsmaterial*. 2. Teubner, 2006

[Ioannou u. Angeli 2013] Ioannou, Ioannis ; Angeli, Charoula: Teaching computer science in secondary education: a technological pedagogical content knowledge perspective. In: *Proceedings of the 8th Workshop in Primary and Secondary Computing Education* (2013), S. 1–7

[Ioannou u. Angeli 2014] Ioannou, Ioannis ; Angeli, Charoula: Examining the effects of an instructional intervention on destabilizing learners' misconceptions about the central processing unit. In: *Proceedings of the 9th Workshop in Primary and Secondary Computing Education on - WiPSCE '14* (2014), S. 93–99

[Iturbide 2012] Iturbide, J. Ángel V.: Identification and removal of misconceptions about greedy algorithms assisted by an interactive system. (2012), S. 1–6

[Jacobs 1998] Jacobs, Bernhard: Aufgaben stellen und Feedback geben / Medienzentrum der Philosophischen Fakultät der Universität Saarbrücken. Saarbrücken, Germany, 1998. – Forschungsbericht. – 1–38 S.

[Jimoyiannis 2011] Jimoyiannis, Athanassios: Using SOLO taxonomy to explore students' mental models of the programming variable and the assignment statement. In: *Themes in Science & Technology Education* 4 (2011), Nr. 2, S. 53–74

[Kaczmarczyk u. a. 2010] Kaczmarczyk, Lisa C. ; Petrick, Elizabeth, R. ; East, J. P. ; Herman, Geoffrey L.: Identifying Student Misconceptions of Programming. In: *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*. Milwaukee, Wisconsin, USA : ACM, 2010, S. 107–111

[Kahney 1983] Kahney, Hank: What do novice programmers know about recursion. (1983), Nr. December, 235–239. `http://dx.doi.org/http://doi.acm.org/10.1145/800045.801618`. – DOI http://doi.acm.org/10.1145/800045.801618. ISBN 0–89791–121–0

[Kansanen 2009] Kansanen, Pertti: The eurious affair of pedagogical content knowledge. In: *Orbis scholae* 3 (2009), Nr. 2, S. 5–18

[Karpierz u. Wolfman 2014] Karpierz, Kuba ; Wolfman, SA: Misconceptions and Concept Inventory Questions for Binary Search Trees and Hash Tables. In:

Proceedings of the 45th ACM technical symposium on Computer science education. Atlanta, GA, USA : ACM, 2014, S. 109–114

[Kattmann 2007] Kattmann, U: Didaktische Rekonstruktion – eine praktische Theorie. In: *Theorien in der biologiedidaktischen Forschung* (2007), S. 93–104

[Kolikant 2001] Kolikant, Yifat Ben-david: Gardeners and Cinema Tickets: High School Students' Preconceptions of Concurrency. In: *Computer Science Education* 11 (2001), Nr. February 2015, S. 221–245

[Krapp u. Weidenmann 2009] Krapp, A ; Weidenmann, B ; Wild, Elke (Hrsg.) ; Möller, Jens (Hrsg.): *Pädagogische Psychologie*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009 (Springer-Lehrbuch)

[Krauss u. a. 2008a] Krauss, Stefan ; Baumert, Jürgen ; Blum, Werner: Secondary mathematics teachers' pedagogical content knowledge and content knowledge: validation of the COACTIV constructs. In: *ZDM Mathematics Education* 40 (2008), Nr. 5, S. 873–892

[Krauss u. a. 2008b] Krauss, Stefan ; Neubrand, Michael ; Blum, Werner ; Baumert, Jürgen ; Brunner, Martin: Die Untersuchung des professionellen Wissens deutscher Mathematik-Lehrerinnen und -Lehrer im Rahmen der COACTIV-Studie Einleitung Das professionelle Wissen von Lehrkräften. 29 (2008), S. 223–258

[Kunter u. Pohlmann 2009] Kunter, Mareike ; Pohlmann, Britta: Lehrer. In: Möller, J. (Hrsg.) ; Wild, E. (Hrsg.): *Einführung in die Pädagogische Psychologie*. Berlin : Springer, 2009, S. 261–282

[Lahtinen u. a. 2005] Lahtinen, Essi ; Ala-Mutka, Kirsti ; Järvinen, Hannu-Matti: A study of the difficulties of novice programmers. In: *ACM SIGCSE Bulletin* 37 (2005), S. 14. http://dx.doi.org/10.1145/1151954.1067453. – DOI 10.1145/1151954.1067453. – ISBN 1595930248

[Lang u. Beauboeuf 2012] Lang, Raymond R. ; Beauboeuf, Theresa: VN-Sim : A Way To Keep Core Concepts in a Crowded Computing Curriculum. In: *Journal of Systemics, Cybernetics & Informatics* 10 (2012), Nr. 1, S. 85–89

[Lewandowski u. a. 2007] Lewandowski, Gary ; Bouvier, Dennis J. ; Mccartney, Robert ; Sanders, Kate ; Simon, Beth: Commonsense Computing (episode 3): Concurrency and Concert Tickets. In: *Proceedings of the third international workshop on Computing education research* (2007), Nr. episode 3, S. 133–144

[Liberman u. a. 2011] Liberman, Neomi ; Beeri, Catriel ; Ben-David Kolikant, Yifat: Difficulties in Learning Inheritance and Polymorphism. In: *ACM Transactions on Computing Education* 11 (2011), Nr. 1, S. 1–23

[Lister u. a. 2004] Lister, Raymond ; Adams, Elizabeth S. ; Fitzgerald, Sue ; Fone, William ; Hamer, John ; Lindholm, Morten ; McCartney, Robert ; Moström, Jan E. ; Sanders, Kate ; Seppälä, Otto ; Simon, Beth ; Thomas, Lynda: A Multi-national Study of Reading and Tracing Skills in Novice Programmers. In: *Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*. New York, NY, USA : ACM, 2004 (ITiCSE-WGR '04), S. 119–150

[Lister u. a. 2006a] Lister, Raymond ; Schulte, Carsten ; Whalley, Jacqueline L. ; Berglund, Anders ; Clear, Tony ; Bergin, Joe ; Garvin-Doxas, Kathy ; Hanks, Brian ; Hitchner, Lew ; Luxton-Reilly, Andrew ; Sanders, Kate: Research perspectives on the objects-early debate. In: *Working group reports on ITiCSE on Innovation and technology in computer science education - ITiCSE-WGR '06* (2006), S. 146

[Lister u. a. 2006b] Lister, Raymond ; Simon, Beth ; Thompson, Errol: Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. In: *Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 2006, S. 118–122

[Ma u. a. 2009] Ma, Linxiao ; Ferguson, John ; Roper, Marc ; Ross, Isla ; Wood, Murray: Improving the mental models held by novice programmers using cognitive conflict and Jeliot visualisations. In: *ACM SIGCSE Bulletin* (2009), 166–170. `http://dl.acm.org/citation.cfm?id=1562931`. ISBN 9781605583815

[Ma u. a. 2007] Ma, Linxiao ; Ferguson, John ; Roper, Marc ; Wood, Murray: Investigating the viability of mental models held by novice programmers. In: *ACM SIGCSE Bulletin* 39 (2007), mar, Nr. 1, 499. `http://dx.doi.org/10.1145/1227504.1227481`. – DOI 10.1145/1227504.1227481. – ISBN 1595933611

[Magenheim u. Schulte 2005] Magenheim, Johannes ; Schulte, Carsten: Erwartungen und Wahlverhalten von Schülerinnen und Schülern gegenüber dem Schulfach Informatik - Ergebnisse einer Umfrage. In: *INFOS* (2005), S. 111–122

[Mandl u. Friedrich 2006] Mandl, Heinz ; Friedrich, Helmut F.: *Handbuch Lernstrategien*. Göttingen : Hogrefe, 2006. – 425 S.

[Maurer 2000] Maurer, Hermann A.: Overflow: Prognosen und Thesen ... nicht nur zum Schmunzeln. In: *Informatik Spektrum* 23 (2000), Nr. 1, S. 51–59

[Mayer 1979] Mayer, Richard E.: A psychology of learning BASIC. In: *Communications of the ACM* 22 (1979), Nr. 11, S. 589–593

[Mayer 1981] Mayer, Richard E.: The Psychology of How Novices Learn Computer Programming. In: *ACM Computing Surveys* 13 (1981), Nr. 1, S. 121–141

[McCall u. Kölling 2014] McCall, D ; Kölling, Michael: Meaningful Categorisation of Novice Programmer Errors. (2014)

[Mccartney u. a. 2009] Mccartney, Robert ; Bouvier, Dennis J. ; Chen, Tzu-Yi ; Sanders, Kate ; Lewandowski, Gary ; Simon, Beth ; Vandegrift, Tammy: Commonsense Computing (episode 5): Algorithm Efficiency and Balloon Testing. In: *Proceedings of the fifth international workshop on Computing education research workshop* (2009), Nr. episode 5, S. 51–62

[Merenluoto 2004] Merenluoto, Kaarina: The Cognitive - Motivational Profiles of Students Dealing With Decimal Numbers and Fractions. In: *Proceedings of the 28th Conference of the International Group for the Psychology of Mathematics Education* Bd. 3, 2004, S. 297–304

[Meyer u. Land 2003] Meyer, Jan H F. ; Land, Ray: Threshold Concepts and Troublesome Knowledge : linkages to ways of thinking and practising within the disciplines. In: *Improving Student Learning – Ten Years On*. 2003, S. 1–16

[Meyer u. Land 2013] Meyer, Jan H F. ; Land, Ray: *Overcoming Barriers to Student Understanding: Threshold Concepts and Troublesome Knowledge*. Taylor & Francis, 2013

[Miller 1988] Miller, Keith: Integrating computer ethics into the computer science curriculum. In: *Computer Science Education* 1 (1988), Nr. 1, S. 37–52. `http://dx.doi.org/10.1080/0899340880010104`. – DOI 10.1080/0899340880010104. – ISBN 0899–3408

[Mindnich u. a. 2008] Mindnich, Anja ; Wuttke, Eveline ; Seifried, Jürgen: Aus Fehlern wird man klug? Eine Pilotstudie zur Typisierung von Fehlern und Fehlersituationen. In: Lankes, Eva-Maria (Hrsg.): *Pädagogische Professionalität als Gegenstand empirischer Forschung*. Münster : Waxmann, 2008, S. 153–164

[Mirani 2015] Mirani, Leo: *Millions of Facebook users have no idea they're using the internet.* 2015

[Modrow 2003] Modrow, Eckart: Pragmatischer Konstruktivismus und fundamentale Ideen als Leitlinien der Curriculumentwicklung. (2003), S. 1–152

[Moosbrugger u. Kelava 2011] Moosbrugger, Helfried ; Kelava, Augustin: *Testtheorie und Fragebogenkonstruktion.* Heidelberg : Springer Medizin Verlag, 2011. – 413 S.

[Mory 2004] Mory, Edna H.: Feedback research revisited. In: *Handbook of research on educational communications and technology.* 2. Mahwah, NJ : Lawrence Erlbaum, 2004, S. 745–784

[Müller 2003] Müller, A.: Fehlertypen und Fehlerquellen beim Physiklernen. Was weiß die Denkpsychologie. In: *Praxis der Naturwissenschaften–Physik in der Schule* 2003 (2003), S. 11–18

[Nussbaum u. Novak 1976] Nussbaum, Joseph ; Novak, Joseph D.: An assessment of children's concepts of the earth utilizing structured interviews. In: *Science Education* 60 (1976), Nr. 4, S. 535–550

[Ohrndorf 2013] Ohrndorf, Laura: A taxonomy of errors for computer science education. In: *IFIP World Computer Congress WCCE 2013.* Torun, 2013

[Ohrndorf 2015] Ohrndorf, Laura: Measuring Knowledge of Misconceptions in Computer Science Education. In: *Proceedings of the eleventh annual International Conference on International Computing Education Research, ICER 2015.* Omaha, NE, USA, 2015, S. 269–270

[Ohrndorf u. Schubert 2013] Ohrndorf, Laura ; Schubert, Sigrid: Measurement of pedagogical content knowledge: students' knowledge and conceptions. In: *Proceedings of the 8th Workshop in Primary and Secondary Computing Education.* Aarhus, Denmark : ACM, 2013

[Ohrndorf u. Schubert 2014] Ohrndorf, Laura ; Schubert, Sigrid: Students' Cognition: Outcomes from an Initial Study with Student Teachers. In: *Proceedings of the 9th Workshop in Primary and Secondary Computing Education.* Berlin, Germany : ACM, 2014, S. 112–115

[Oldenburg 2012] Oldenburg, Reinhard: *Mathematische Algorithmen im Unterricht - Mathematik aktiv erleben durch Programmieren.* Vieweg und Teubner, 2012. – 176 S.

[Oldham 2005] Oldham, Joseph D.: What happens after Python in CS1? In: *Journal of Computing Sciences in Colleges* 20 (2005), Nr. 6, S. 7–13

[Oser u. Spychinger 2005] Oser, Fritz ; Spychinger, M.: *Lernen ist schmerzhaft. Zur Theorie des Negativen Wissens und zur Praxis der Fehlerkultur*. Weinheim : Beltz, 2005

[Özdener 2008] Özdener, Nesrin: A comparison of the misconceptions about the time-efficiency of algorithms by various profiles of computer-programming students. In: *Computers and Education* 51 (2008), S. 1094–1102

[Pamplona u. a. 2013] Pamplona, Sonia ; Medinilla, Nelson ; Flores, Pamela: Exploring misconceptions of operating systems in an online course. In: *Proceedings of the 13th Koli Calling International Conference on Computing Education Research - Koli Calling '13*. Koli, Finland : ACM, 2013, S. 77–86

[Paul u. Vahrenhold 2013] Paul, Wolfgang ; Vahrenhold, Jan: Hunting High and Low: Instruments to Detect Misconceptions Related to Algorithms and Data Structures. In: *The 44th ACM Technical Symposium on Computer Science Education, SIGCSE '13*. Denver, Colorado, USA, 2013, S. 29–34

[Pea 1986] Pea, Roy D.: Language-Independent Conceptual Bugs in Novice Programming. In: *Journal of Educational Computing Research* 2 (1986), Nr. 1, S. 25–36

[Pham u. a. 2011] Pham, Manh C. ; Klamma, Ralf ; Jarke, Matthias: Development of computer science disciplines: a social network analysis approach. In: *Social Network Analysis and Mining* 1 (2011), Nr. 4, S. 321–340

[Porst 2009] Porst, Rolf ; Sahner, Heinz (Hrsg.) ; Bayer, Michael (Hrsg.) ; Sackmann, Reinhold (Hrsg.): *Fragebogen - Ein Arbeitsbuch*. 2. Auflage. Heidelberg : Springer-Verlag, 2009. – 195 S.

[Porter u. a. 2013] Porter, Leo ; Garcia, Saturnino ; Tseng, Hung-Wei ; Zingaro, Daniel: Evaluating student understanding of core concepts in computer architecture. In: *Proceedings of the 18th ACM conference on Innovation and technology in computer science education - ITiCSE '13* (2013), 279. `http://dx.doi.org/10.1145/2462476.2462490`. – DOI 10.1145/2462476.2462490. – ISBN 9781450320788

[Prediger u. Wittmann 2009] Prediger, Susanne ; Wittmann, Gerald: Aus Fehlern lernen - (wie) ist das moeglich? In: *Praxis der Mathematik in der Schule* 27 (2009), Nr. 1

[Radatz 1979] Radatz, Hendrik: Error Analysis in Mathematics Education. In: *Journal for Research in Mathematics Education* 10 (1979), Nr. 3, S. 163–172

[Ramalingam u. Wiedenbeck 1997] Ramalingam, Vennila ; Wiedenbeck, Susan: An empirical study of novice program comprehension in the imperative and object-oriented styles. In: *Papers presented at the seventh workshop on Empirical studies of programmers - ESP '97* (1997), 124–139. `http://dx.doi.org/10.1145/266399.266411`. – DOI 10.1145/266399.266411. ISBN 0897919920

[Rand 1971] Rand, W M.: Objective criteria for the evaluation of clustering methods. In: *Journal of the American Statistical Association* 66 (1971), Nr. 336, S. 846–850

[Reiss u. Hammer 2010] Reiss, Kristina ; Hammer, Christoph: *Grundlagen der Mathematikdidaktik*. Birkhäuser, 2010. – 147 S.

[Reusser 1999] Reusser, Kurt: Schülerfehler: die Rückseite des Spiegels. In: Althof, W. (Hrsg.): *Fehlerwelten*. Leske & Budrich, 1999, S. 213–248

[Rountree u. a. 2013] Rountree, Janet ; Robins, Anthony ; Rountree, Nathan: Elaborating on threshold concepts. In: *Computer Science Education* 23 (2013), Nr. February 2015, S. 265–289

[Rowe u. Smaill 2008] Rowe, Gerard ; Smaill, Chris: Development of an electromagnetics course concept inventory. (2008)

[Ruf u. a. 2014] Ruf, Alexander ; Mühling, Andreas ; Hubwieser, Peter: Scratch vs . Karel – Impact on Learning Outcomes and Motivation. In: *Proceedings of the 9th Workshop in Primary and Secondary Computing Education*. Aarhus, Denmark, 2014, S. 50–59

[Saeli u. a. 2011] Saeli, Mara ; Perrenet, Jacob ; Jochems, Wim M. ; Zwaneveld, Bert: Teaching Programming in Secondary School: a Pedagogical Content Knowledge Perspective. In: *Informatics in Education* 10 (2011), Nr. 1, S. 73–88

[Sanders u. a. 2012] Sanders, Kate ; Boustedt, Jonas ; Eckerdal, Anna ; Moström, Jan E. ; Mccartney, Robert ; Thomas, Lynda ; Zander, Carol: Threshold Concepts and Threshold Skills in Computing Categories and Subject

Descriptors. In: *Proceedings of the ninth annual international conference on International computing education research*. Auckland, New Zealand : ACM, 2012, S. 23–30

[Sanders u. Thomas 2007] Sanders, Kate ; Thomas, Lynda: Checklists for Grading Object-Oriented CS1 Programs: Concepts and Misconceptions. In: *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*. Dundee, Scotland, United Kingdom, 2007, S. 166–170

[Sarkar u. a. 2013] Sarkar, Amitrajit ; Lance, Michael ; Lopez, Mike ; Oliver, Robert ; Xu, Luofeng: Student mistakes in an introductory programming course. In: *citrenz.ac.nz* (2013). `http://www.citrenz.ac.nz/conferences/2013/pdf/2013CITRENZ{_}1{_}Sarkar18-Programming.pdf`

[Schubert u. Schwill 2011] Schubert, Sigrid ; Schwill, Andreas: *Didaktik der Informatik*. 2. Auflage. Berlin, Heidelberg : Springer, 2011. – 418 S.

[Schulte u. Bennedsen 2006] Schulte, Carsten ; Bennedsen, Jens: What do teachers teach in introductory programming? In: *Proceedings of the 2006 international workshop on Computing education research - ICER '06* (2006), S. 17

[Schwill] Schwill, Andreas: Fundamentale Ideen in Mathematik und Informatik.

[Schwill 1993] Schwill, Andreas: Fundamentale ideen der informatik. In: *Zentralblatt für Didaktik der Mathematik* (1993), S. 1–30

[Schwill 1994] Schwill, Andreas: Fundamental ideas of computer science. In: *Bulletin-European Association for Theoretical Computer Science* 53 (1994), S. 274. ISBN 0252–9742

[Seel 2003] Seel, Norbert: *Psychologie des Lernens: Lehrbuch für Pädagogen und Psychologen*. E. Reinhardt, 2003

[Seifried u. a. 2010] Seifried, Jürgen ; Türling, Janosch M. ; Wuttke, Eveline: Professionelles Lehrerhandeln - Schülerfehler erkennen und für Lernprozesse nutzen. In: Warwas, Julia (Hrsg.) ; Sembill, Detlef (Hrsg.): *Schule zwischen Effizienzkriterien und Sinnfragen*. Baltmannsweiler : Schneider Verlag Hohengehren GmbH, 2010, S. 137–156

[Sekiya u. Yamaguchi 2013] Sekiya, Takayuki ; Yamaguchi, Kazunori: Tracing Quiz Set to Identify Novices' Programming Misconceptions. In: *Proceedings of the 13th Koli Calling International Conference on Computing Education Research - Koli Calling '13*. Koli, Finland : ACM, 2013, S. 87–95

[Shinners-Kennedy u. Fincher 2013] Shinners-Kennedy, Dermot ; Fincher, Sally: Identifying threshold concepts: From dead end to a new direction. In: *Proceedings of the ninth annual international ACM conference on International computing education research* (2013), S. 9–18

[Shrout u. Fleiss 1979] Shrout, Patrick E. ; Fleiss, Joseph L.: Intraclass correlations: uses in assessing rater reliability. In: *Psychological bulletin* 86 (1979), Nr. 2, S. 420–428

[Shulman 1986] Shulman, Lee S.: Those Who Understand: Knowledge Growth in Teaching. In: *Educational Researcher* 15 (1986), Nr. 2, S. 4–14

[Shulman 1987] Shulman, Lee S.: Knowledge and Teaching: Foundations of the New Reform. In: *Harvard Educational Review* 57 (1987), Nr. 1, S. 1–21

[Simon 2011] Simon: Assignment and sequence: why some students can't recognise a simple swap, 2011, S. 10–15

[Simon u. a. 2008] Simon, Beth ; Bouvier, Dennis ; Chen, Tzu-Yi ; Lewandowski, Gary ; McCartney, Robert ; Sanders, Kate: Common sense computing (episode 4): debugging. In: *Computer Science Education* 18 (2008), Nr. February 2015, S. 117–133

[Simon u. a. 2006] Simon, Beth ; Chen, Tzu-Yi ; Lewandowski, Gary ; Mccartney, Robert ; Sanders, Kate: Commonsense computing: what students know before we teach (episode 1: sorting). In: *Proceedings of the second international workshop on Computing education research - ICER '06*. Canterbury, England, UK, 2006, S. 29–40

[Sirkiä u. Sorva 2012] Sirkiä, Teemu ; Sorva, Juha: Exploring Programming Misconceptions. In: *Proceedings of the 12th Koli Calling International Conference on Computing Education Research*. Tahko, FInland : ACM, 2012, S. 19–28

[Sleeman 1984] Sleeman, D.: Pascal and High-School Students: A Study of Misconceptions. (1984). ISBN 0934387001

[Smith u. a. 1993] Smith, John P. ; DiSessa, Andrea A. ; Roschelle, Jeremy: Misconceptions reconceived: A constructivist analysis of knowledge in transition. In: *The Journal of the Learning Sciences* 3 (1993), Nr. 517, S. 115–163

[Smith u. a. 1994] Smith, John P. I. ; DiSessa, Andrea A. ; Roschelle, Jeremy: Misconceptions Reconceived: A Constructivist Analysis of Knowledge in Transition. In: *Journal of the Learning Sciences* 3 (1994), Nr. 2, S. 115–163

[Sorva 2007] Sorva, Juha: Students' understandings of storing objects. In: *Proceedings of the Seventh Baltic Sea Conference on ...* (2007), S. 127–135

[Sorva 2008] Sorva, Juha: The same but different - Students' understandings of primitive and object variables. In: *Proceedings of the 8th International Conference on Computing Education Research - Koli '08* (2008), 5. http://dx.doi.org/10.1145/1595356.1595360. – DOI 10.1145/1595356.1595360. ISBN 9781605583853

[Sorva 2010] Sorva, Juha: Reflections on threshold concepts in computer programming and beyond. In: *Proceedings of the 10th Koli Calling International Conference on Computing Education Research - Koli Calling '10* (2010), S. 21–30

[Sorva 2012] Sorva, Juha: *Visual Program Simulation in Introductory Programming Education*, Aalto University, Diss., 2012

[Sorva 2013] Sorva, Juha: Notional machines and introductory programming education. In: *ACM Transactions on Computing Education* 13 (2013), jun, Nr. 2, 1–31. http://dx.doi.org/10.1145/2483710.2483713. – DOI 10.1145/2483710.2483713. – ISSN 19466226

[Spohrer u. Soloway 1986a] Spohrer, James C. ; Soloway, Elliot: Alternatives to construct-based programming misconceptions. In: *ACM SIGCHI Bulletin*. Boston, Massachusetts, USA, 1986, S. 183–191

[Spohrer u. Soloway 1986b] Spohrer, James C. ; Soloway, Elliot: Novice mistakes: are the folk wisdoms correct? In: *Communications of the ACM* 29 (1986), Nr. 7, S. 624–632

[Sudol u. Jaspan 2010] Sudol, Leigh A. ; Jaspan, Ciera: Analyzing the Strength of Undergraduate Misconceptions About Software Engineering Categories and Subject Descriptors. In: *Proceedings of the Sixth International Workshop on Computing Education Research - ICER'10*. Aarhus, Denmark : ACM, 2010, S. 31–39

[Tao u. Gunstone 1999] Tao, Ping-Kee ; Gunstone, Richard F.: The process of conceptual change in force and motion during computer-supported physics instruction. In: *Journal of Research in Science Teaching* 36 (1999), Nr. 7, S. 859–882

[Taylor u. a. 2014] Taylor, C. ; Zingaro, D. ; Porter, L. ; Webb, K.C. ; Lee, C.B. ; Clancy, M.: Computer science concept inventories: past and future. In: *Computer Science Education* 24 (2014), Nr. 4, S. 253–276

[Taylor 2006] Taylor, Charlotte ; Land, Ray (Hrsg.) ; Meyer, Jan (Hrsg.): *Threshold Concepts in Biology: do they fit the definition?* Taylor & Francis, 2006. – 87–99 S.

[Teif u. Hazzan 2006] Teif, Mariana ; Hazzan, O: Partonomy and taxonomy in object-oriented thinking: junior high school students' perceptions of object-oriented basic concepts. In: *ACM SIGCSE Bulletin* 38 (2006), Nr. 4, S. 55–60

[Tew 2010] Tew, Allison E.: Assessing Fundamental Introductory Computing Concept Knowledge in a Language Independent Manner Assessing Fundamental Introductory Computing Concept Knowledge. (2010), Nr. December

[Tew u. Guzdial 2010] Tew, Allison E. ; Guzdial, Mark: Developing a Validated Assessment of Fundamental CS1 Concepts. In: *Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10*. Milwaukee, Wisconsin, USA : ACM, 2010, S. 97

[Tew u. Guzdial 2011] Tew, Allison E. ; Guzdial, Mark: The FCS1 : A Language Independent Assessment of CS1 Knowledge. In: *Symposium A Quarterly Journal In Modern Foreign Literatures* (2011), S. 111–116

[Tew u. a. 2005] Tew, Allison E. ; McCracken, W. M. ; Guzdial, Mark: Impact of alternative introductory courses on programming concept understanding. In: *Proceedings of the 2005 international workshop on Computing education research - ICER '05*. Seatlle, Washington, USA : ACM, 2005, S. 25–35

[The Joint Task Force on Computing Curricula 2001] The Joint Task Force on Computing Curricula: Computing Curricula 2001 Computer Science. In: *IEEE-CS, ACM. Final Report* (2001)

[The Joint Task Force on Computing Curricula u. a. 2004] The Joint Task Force on Computing Curricula ; Society, IEEE C. ; Machinery, Association for C.: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering. 2004 (0229748). – Forschungsbericht

[Thomasson u. a. 2006] Thomasson, Benjy ; Ratcliffe, Mark ; Thomas, Lynda: Identifying novice difficulties in object oriented design. In: *ACM SIGCSE Bulletin* (2006), 28–32. `http://dl.acm.org/citation.cfm?id=1140135`. ISBN 1595930558

[Thompson 2007] Thompson, Errol: Holistic assessment criteria - Applying SOLO to programming projects. In: *Conferences in Research and Practice in Information Technology Series* 66 (2007), S. 155–162

[Tirronen 2014] Tirronen, Ville: Study on Difficulties and Misconceptions With Modern Type Systems. In: *Iticse* (2014), S. 303–308

[Treagust 1988] Treagust, David F.: Development and use of diagnostic tests to evaluate students' misconceptions in science. In: *International Journal of Science Education* 10 (1988), Nr. 2, S. 159–169

[Tucker 1996] Tucker, Allen B Et A.: Strategic Directions in Computer Science Education. In: *ACM Computing Surveys* 28 (1996), Nr. 4, S. 836–845

[Turner u. a. 2008] Turner, Scott A. ; Quintana-Castillo, Ricardo ; Pérez-Quiñones, Manuel A. Edwards, Stephen H.: Misunderstandings about object-oriented design: experiences using code reviews. In: *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*. Portland, Oregon, USA, 2008, S. 97–101

[Vahrenhold u. Paul 2014] Vahrenhold, Jan ; Paul, Wolfgang: Developing and validating test items for first-year computer science courses. In: *Computer Science Education* 24 (2014), Nr. 4, 304–333. `http://dx.doi.org/10.1080/08993408.2014.970782`. – DOI 10.1080/08993408.2014.970782. – ISSN 0899–3408

[Vandegrift u. a. 2010] Vandegrift, Tammy ; Bouvier, Dennis J. ; Chen, Tzu-Yi ; Lewandowski, Gary ; Mccartney, Robert ; Simon, Beth: Commonsense Computing (episode 6): Logic is Harder than Pie. In: *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*. Koli, Finland : ACM, 2010, S. 76–85

[Venables u. a. 2009] Venables, Anne ; Tan, Grace ; Lister, Raymond: A closer look at tracing, explaining and code writing skills in the novice programmer. In: *... of the fifth international workshop on ...* (2009), Nr. 2008, S. 117–128

[Vosniadou u. Brewer 1992] Vosniadou, Stella ; Brewer, William F.: Mental Models of the Earth: A Study of Conceptual Change in Childhood. In: *Cognitive Psychology* 24 (1992), S. 535–585

[Webb u. Taylor 2014] Webb, Kevin C. ; Taylor, Cynthia: Developing a pre- and post-course concept inventory to gauge operating systems learning. In: *Proceedings*

of the 45th ACM technical symposium on Computer science education - SIGCSE '14. Atlanta, GA, USA : ACM, 2014, S. 103–108

[White 2009] White, Alan L.: A Revaluation of Newman's Error Analysis. In: *MAV Annual Conference 2009* Bd. 3, 2009, 249–257

[Wilhelm 2005] Wilhelm, Thomas: Mechanik bei bayerischen Elftklässlern-Ergebnisse beim Test Force Concept Inventory in herkömmlichen Klassen und im Würzburger Kinematik-/Dynamikunterricht. In: *Physik und Didaktik in Schule und Hochschule* 4 (2005), Nr. 2, S. 47–56

[Wu u. a. 1998] Wu, Cheng-Chih ; Dale, Nell B. ; Bethel, Lowell J.: Conceptual models and cognitive learning styles in teaching recursion. In: *Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education - SIGCSE '98* (1998), S. 292–296

[Yehezkel u. a. 2001] Yehezkel, Cecile ; Ben-ari, Mordechai ; Dreyfus, Tommy: Inside the Computer: Visualization and Mental Models. In: *Third Program VisualizationWorkshop*, 2001, S. 1–4

[Zendler u. Spannagel 2008] Zendler, Andreas ; Spannagel, Christian: Empirical foundation of central concepts for computer science education. In: *Journal on Educational Resources in Computing* 8 (2008), Nr. 2, S. 1–15

[Zendler u. a. 2008] Zendler, Andreas ; Spannagel, Christian ; Klaudt, Dieter: Process as content in computer science education: empirical determination of central processes. In: *Computer Science Education* 18 (2008), Nr. 4, S. 231–245

[Zendler u. a. 2010] Zendler, Andreas ; Spannagel, Christian ; Klaudt, Dieter: Experimentelle Untersuchung zur Grundlegung einer forschungsbasierten Informatiklehrerausbildung. In: *Notes on Educational Informatics - Section A: Concepts and Techniques* 6 (2010), Nr. 1, S. 25–43

[Zendler u. a. 2011] Zendler, Andreas ; Spannagel, Christian ; Klaudt, Dieter: Marrying Content and Process in Computer Science Education. In: *IEEE Transactions on Education* 54 (2011), Nr. 3, S. 387–397