Beitrag erschienen in:

Soll, Marcus; Kobras, Louis; Lagod, Christian: Gibt es ein Problemfach im Informatikstudium?. In (Desel, Jörg; Opel, Simone, Hrsg.): Hochschuldidaktik Informatik (HDI) 2023, 10. Fachtagung des GI-Fachbereichs Informatik, 13.–14. September 2023 in Aachen, Bd. 14, Commentarii informaticae didacticae (CID), Potsdam: Universitätsverlag Potsdam, S. 65–86, 2025, DOI: 10.25932/publishup-68776

Gibt es ein Problemfach im Informatikstudium?

Marcus Soll[1], Louis Kobras[1] und Christian Lagod[2]

Abstract: In diesem Beitrag soll die Frage beantwortet werden, ob es einzelne Fächer im Informatikstudium gibt, die besonders schwer sind – sogenannte Problemfächer. Dazu werden zuerst vorhandene Daten und Studien ausgewertet. Auf Grundlage von Studien über Studienabbrecher*innen in der Informatik, Wahrnehmung von Dozent*innen und Student*innen sowie Notenspiegeln werden zwei Bereiche identifiziert, die potentiell ein Problemfach darstellen können: mathematisch-theoretische Fächer sowie programmiernahe Fächer. Durch eine Untersuchung relevanter Literatur kann bestätigt werden, dass es sich bei den mathematisch-theoretischen Fächern mit hoher Wahrscheinlichkeit um Problemfächer handelt, die programmiernahen Fächer allerdings nicht als Problemfach betrachtet werden können. Die Reflexion der Ergebnisse enthält die Diskussion von der Fachtagung Hochschuldidaktik der Informatik HDI 2023.

Keywords: Curriculum Analytics; Informatikstudium; Problemfächer; Studienverlaufsanalyse

1 Einleitung

In vielen persönlichen Gesprächen – sowohl mit Student*innen als auch mit Dozent*innen – haben die Autoren gehört, dass es bestimmte Fächer im Informatikstudium gibt, die als besonders schwer gelten und mit denen die Student*innen besonders viele Probleme haben. Diese Einstellung scheint auch in der Literatur weit verbreitet zu sein; so haben sich Studien, die sich mit Schwierigkeiten und

1 NORDAKADEMIE gAG Hochschule der Wirtschaft, Köllner Chaussee 11, 25337 Elmshorn, Deutschland, marcus.soll@nordakademie.de https://orcid.org/0000-0002-6845-9825 | louis.kobras@nordakademie.de https://orcid.org/0000-0003-4855-2878

2 Universität Hamburg, Fachbereich Informatik, Vogt-Kölln-Straße 30, 22527 Hamburg, Deutschland, 9lagod@informatik.uni-hamburg.de

Herausforderungen von Student*innen im Informatikstudium befassen, häufig auf ein als subjektiv schwierig angesehenes Fach konzentriert, z. B. [Sa21].

Die bisherige Meinung, welche Fächer im Informatikstudium schwierig oder problematisch sind, basiert daher häufig auf Anekdoten. Dieses Problem wurde bereits mehrfach in der Literatur angesprochen [BC07; WL14] und zumindest für die Einführung in Programmierung haben dieselben Autoren auch erste Studien veröffentlicht, die die Probleme in den Programmierfächern an Hand von Daten[1] nachweisen [BC07; WL14].

In dieser Arbeit soll daher die Frage angegangen werden, ob es tatsächlich ein Problemfach in der Informatik gibt. Ein Problemfach wird hier definiert als ein Fach, bei dem ein großer Anteil an Student*innen Probleme haben, unabhängig von Dozent*innen oder institutionsspezifischen Herausforderungen. Nach bestem Wissen der Autoren gibt es dabei noch keine Studie, die diese Frage direkt adressiert. Stattdessen sollen in dieser Arbeit verschiedene Daten zusammengetragen werden, die die Frage nach einem Problemfach beantworten können.

Die Beantwortung dieser Frage kann durchaus Konsequenzen dafür haben, wie wir in Zukunft den Fokus auf die Verbesserung der Informatik-Studiengänge richten. Sollte es ein oder mehrere Problemfächer geben, so würde es sich lohnen, sich auf diese zu konzentrieren (wie es in der Vergangenheit passiert ist [Sa21]). Gibt es aber kein Problemfach, so könnte sich die Forschung eher auf systematische Probleme im Informatikstudium fokussieren. Solche systematischen Probleme könnten unter anderem problematische Studienbedingungen sein oder Probleme außerhalb des Studiums, die zu einem Studienabbruch führen. Solche Studien nach systematischen Problemen unabhängig von einem Fach findet man zum Beispiel bei Neugebauer, Heublein und Daniel [NHD19] und Behr et al. [Be21].

2 Hinweise auf bestehende Problemfächer

Um mögliche Problemfächer zu finden, werden in diesem Abschnitt zunächst verschiedene Studien aus unterschiedlichen Bereichen betrachtet. Basierend auf den Ergebnissen dieser Studien werden dann in Abschnitt 3 die identifizierten potentiellen Problemfächer genauer betrachtet.

1 Es handelt sich bei beiden um Bestehensquoten. Bennedsen und Caspersen [BC07] haben dafür eine Umfrage an Institutionen versendet, Watson und Li [WL14] nutzen vorhandene Daten aus Literatur.

2.1 Studienabbruchgründe

Eine erste Annäherung an Problemfächer kann anhand von Studienabbrecher*innen – und weiter gedacht in Studienberatungen – betrachtet werden. Sollte es ein oder mehrere Problemfächer geben, so wäre die Erwartung, dass dieses Fach zu vielen Studienabbrüchen führt, was sich auch in entsprechenden Studienberatungen wiederfinden sollte, so die abbrechenden Studierenden denn ein entsprechendes Angebot wahrnehmen. Dementsprechend soll in diesem Abschnitt Literatur zum Thema Studienabbruch in der Informatik betrachtet werden.

Böttcher et al. [Bö21] berichten ihre Erfahrungen in der Studienberatung in der Fakultät für Informatik und Mathematik an der Hochschule München. Aus ihrer „Hypothese 4“ lässt sich ableiten, dass insbesondere Mathematik und programmiernahe Fächer ein starker Prädiktor für einen Studienabbruch sind. Diese Fächer können daher laut dieser Studie als problematisch angesehen werden. Hierbei ist allerdings zu beachten, dass an der Hochschule München ein Nichtbestehen (oder Nichtantreten) bei allen Mathematik-Modulen im 1. Semester automatisch zu einem Nichtbestehen von Mathematikmodulen im 2. Semester führt, wodurch die Mathematikmodule einen deutlich stärkeren Einfluss auf das Studium haben könnten als andere Fächer.

Weitere aktuelle Daten sind leider schwer zu bekommen. Stattdessen muss hier auf ältere Daten zurückgegriffen werden. So hat Jonkmann [Jo05] Informatikstudent*innen befragt, die im Wintersemester 2004/05 eingeschrieben und mindestens im zweiten Semester waren. Hier wurden als Gründe Probleme beim Studieneinstieg, ein zu schweres Studium, Desinteresse bzw. falsche Vorstellungen sowie finanzielle Aspekte genannt. Insbesondere wurde auch eine zu hohe Theorielastigkeit und zu viel Mathematik als ein Grund für Abbrüche genannt.

Holdt, Schneider und Wagner [HSW06] berichten von ihrer Erhebung von Daten aus den Jahren 2003 bis 2006 durch Befragungen von Studienanfänger*innen sowie Daten aus Prüfungen und telefonischer Befragung von Studienabbrecher*innen. Als schwierige Fächer wurden die theoretische Informatik, Stochastik sowie die Halbleiterschaltungstechnik ermittelt. Die wichtigsten Abbruchgründe setzen sich hier aus hohen Anforderungen (insbesondere in Mathematik), falschen Vorstellungen (insbesondere einem fehlenden Praxisbezug) sowie finanziellen Problemen zusammen.

Tab. 1: Wichtigste Abbruchgründe zusammengefasst aus verschiedenen Studien. Fächer, die mindestens die Hälfte der Studien als Abbruchgrund sehen, sind hervorgehoben.

Abbruchgrund	[Bö21]	[Jo05]	[HSW06]	[He10]
fachlich: Theoretische Informatik		X	X	
fachlich: Mathematik	X	X	X	
fachlich: Programmieren	X			
fachlich: Elektrotechnik			X	
Desinteresse / falsche Vorstellungen		X	X	X
finanzielle Aspekte		X	X	X
hohe Anforderungen		X	X	X
problematische Studienbedingungen				X
Probleme beim Studieneinstieg		X		

Heublein et al. [He10] haben bundesweit im Jahr 2007/08 exmatrikulierte Student*innen befragt. Für das Fach Informatik konnten sie feststellen, dass die wichtigsten Gründe für einen Studienabbruch hohe Leistungsanforderungen, falsche Vorstellungen, finanzielle Gründe sowie problematische Studienbedingungen waren.

Insgesamt lässt sich feststellen, dass es eine Vielzahl von Gründen gibt, warum Student*innen ihr Studium in der Informatik abbrechen. Eine Übersicht aller vorgestellten Studien findet sich in Tabelle 1. Zunächst sticht hervor, dass nicht-fachliche Gründe einschlägig für die Abbruchquoten sind oder zumindest vor wenigen Jahren noch waren, wie [He10; HSW06; Jo05] erhoben haben. Diese sind aber nicht Gegenstand dieser Studie. Mit dem Blick auf die Problemfächer fällt hier vor allem **der Bereich der theoretischen Informatik** sowie der **Mathematik** auf. Dabei gibt es aber zwei Aspekte, die beachtet werden müssen: zum einen, dass viele Studien relativ alte Daten benutzen und daher nicht unbedingt die momentane Situation widerspiegeln; zum anderen, dass der Anteil der Probleme durch einzelne Fächer insgesamt doch beschränkt ist (z. B. bei Heublein et al. [He10, S. 155 f.] ist der ganze Bereich der „*Leistungsprobleme*“ – inklusive möglicher Problemfächer – nur 25 % groß).

2.2 Wahrnehmung von Dozent*innen / Student*innen

Ein weiterer Indikator für ein Problemfach entsteht durch die Wahrnehmung von Dozent*innen und Student*innen. Wenn beide Gruppen ein Fach als besonders schwierig oder problematisch wahrnehmen, dann ist die Chance groß, dass dieses Fach tatsächlich ein Problemfach ist. Hier sollte man trotzdem Obacht walten lassen, da durchaus auch Vorurteile in die Wahrnehmung einfließen. Es gibt zwei Studien, die wir hier betrachten können: Bender et al. [Be23] haben Dozent*innen befragt und Soll und Kobras [SK22] haben eine Nachfolgestudie mit Student*innen durchgeführt.

In ihrer Studie haben Bender et al. [Be23] Dozent*innen deutschlandweit befragt, welche allgemeinen bzw. fachbezogenen Fähigkeiten und Kenntnisse sie von Studienanfänger*innen erwarten. Für die Betrachtung von Problemfächern ist jedoch die dritte Frage der Studie nach problembehafteten Bereichen relevant. Hier konnten die Autor*innen der Studie feststellen, dass die Dozent*innen eher theoretische Bereiche als problematisch ansehen (siehe Tabelle 2 für eine genaue Übersicht).

Es sollte beachtet werden, dass Dozent*innen allerdings aufgrund eigener Erfahrungen und Wahrnehmungen grundsätzlich kritischer auf bestimmte Kurse und die Leistungen der Student*innen in ebenjenen Kursen sehen könnten und dadurch die Antworten von der Realität abweichen (z. B. durch *Confirmation Bias* [Ni98] oder *Selection Bias* [Be10]).

Ein anderes Bild ergibt sich, wenn Student*innen befragt werden. Soll und Kobras [SK22] konnten dabei feststellen, dass es nach Ansicht von Student*innen überhaupt keine problembehafteten Bereiche gibt. Stattdessen werden über ein Drittel aller Bereiche als explizit nicht problembehaftet angesehen (siehe Tabelle 2 für eine genaue Übersicht). Im Gegensatz zu der Befragung von Dozent*innen muss davon ausgegangen werden, dass Student*innen hier die problembehafteten Bereiche unterschätzen (*Selection Bias* [Be10]). Zum einen werden Student*innen, die ein Fach nicht bestanden haben, nicht mehr Informatik studieren, zum anderen haben an der Umfrage sehr viele Student*innen aus den ersten Semestern teilgenommen, die einige Fächer noch nicht kennen. Zudem kann es passieren, dass ein*e Student*in die Einstellung von anderen übernimmt, wenn alle Student*innen im Umfeld ein Fach als schwierig/einfach bezeichnen (*Social-Desirability Bias* [KB00]).

Tab. 2: Problembehaftete Bereiche (p) und nicht-problemhafte Bereiche (n) nach Bender et al. [Be23] und Soll und Kobras [SK22]. Ein Bereich hier wird hier als problembehaftet markiert, wenn mindestens 50 % der Teilnehmer zustimmen.

Bereich	[Be23]	[SK22]
Automatentheorie	p	
Binärsystem	n	n
Datenanalysesprachen		n
Datenbanken		n
Dokumentenbeschreibungssprachen	n	n
Formale Logik	p	
Formale Sprachen	p	
Kenntnisse über elementare Algorithmen		
Kenntnisse Zusammenhang Informatik & Gesellschaft	n	n
Lesen von formalisierter Schreibweise	p	
Mengenlehre		
Modellierung von Problemen	p	
Programmiersprachen	p	
Rechneraufbau/Hardware		n
Rechnernetzwerke		
Schreiben von formalisierter Schreibweise	p	
Umgang mit handelsüblicher Software	n	n
Umgang mit Logarithmen	p	
Umgang mit (mehreren) Betriebssystemen		n

Insgesamt lässt sich festhalten, dass es keinen direkten Widerspruch zwischen der Sicht der Student*innen und der Sicht der Dozent*innen gibt, d. h. es gibt kein Fach, das von der einen Gruppe als problembehaftet angesehen wird und von der anderen Gruppe explizit nicht. Trotzdem lässt sich die Frage hier nicht endgültig beantworten, auch wenn die Antworten der Dozent*innen darauf hindeuten, dass die Probleme eher im **theoretischen Bereich** und vielleicht auch bei **Programmierfächern** liegen könnten.

2.3 Bestehensquoten / Notenspiegel der Universität Hamburg

Einen Datensatz, an dem sich ein Problemfach ausfindig machen lassen könnte, stellen Noten dar. Zwar kann die Aussagekraft von Noten und Bestehensquoten durchaus angezweifelt werden [Ts19], dafür sind die Noten als quantitative Daten an jeder Hochschule verfügbar. Da die Notenspiegel von Hochschulen meist nicht öffentlich verfügbar sind, wurden Anfragen nach dem hamburgischen Transparenzgesetz an verschiedene öffentliche Hochschulen in Hamburg gestellt. Inhalt der Anfrage waren die Noten der Pflichtmodule der ersten Semester eines Informatikstudiums. Als einzige Hochschule hat sich die Universität Hamburg hier positiv zurückgemeldet.

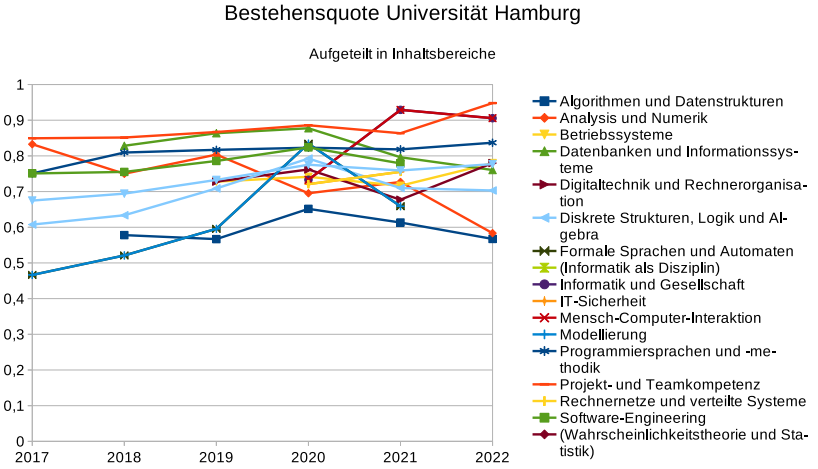
Die Informatikveranstaltungen und deren Prüfungen finden innerhalb von Modulen statt. Da diese Module von Hochschule zu Hochschule sehr variieren[2], werden diese Module bestimmten Fächern/Bereichen zugeordnet, damit eine strukturelle/inhaltliche Betrachtung erst ermöglicht wird. Dafür werden die „Empfehlungen für Bachelor- und Masterprogramme im Studienfach Informatik an Hochschulen“ der Zukunft [Zu16] verwendet[3]. Die Ergebnisse finden sich in Abb. 1.

Dabei zeigt sich, dass die Bestehensquoten sowie die Durchschnittsnoten in den verschiedenen Jahren starken Schwankungen unterliegen. So ist es schwierig, hier eindeutige Bereiche zu erkennen, die in allen Jahren schwierig sind. Als Beispiel: Der Bereich *Modellierung* bzw. *Formale Sprachen* hat in den Jahren 2017 und 2018 die schlechteste Bestehensquote, im Jahr 2020 jedoch eine der besten Bestehensquoten.

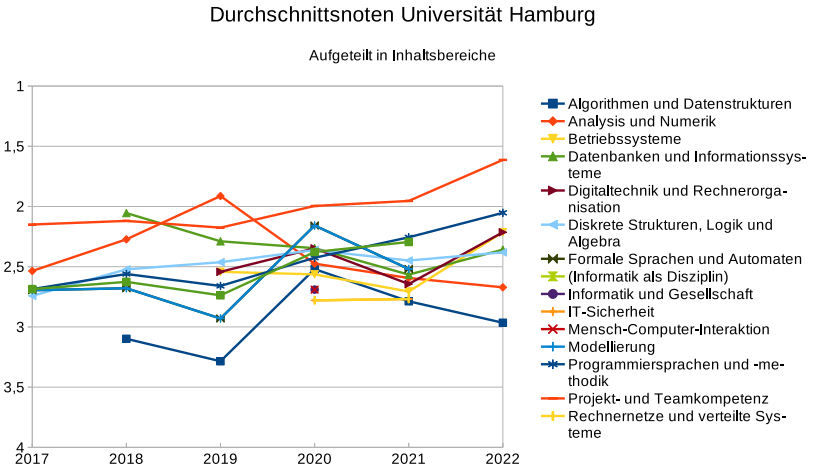
Einzige Ausnahme bildet hier der Bereich *Algorithmen und Datenstrukturen*. Dieser ist in fast allen Jahren das Fach mit sowohl der schlechtesten Bestehensquote als auch mit dem schlechtesten Notendurchschnitt. Dies könnte durchaus auf ein Problemfach hindeuten. Da dies aber der einzige Bereich ist und verwandte Bereiche (z. B. *Diskrete Strukturen, Logik und Algebra* oder *Modellierung*) diesem Trend nicht folgen, muss hier in Betracht gezogen werden, dass es zu einem Abweichen aufgrund anderer Parameter gekommen sein kann.

2 Jede Hochschule hat ihre Veranstaltungen wegen Bologna [Jo99] modularisiert und nennt ihre Module anders. Zudem ist deren Größe und Inhalt je nach Hochschule sehr verschieden.

3 Die Bereiche *Informatik als Disziplin* sowie *Wahrscheinlichkeitstheorie und Statistik* sind dabei bei der Universität Hamburg nicht als Pflichtmodul in den ersten Semestern enthalten.



(a) Bestehensquote



(b) Durchschnittsnoten

Abb. 1: Bestehensquote und Durchschnittsnoten der Pflichtfächer der Universität Hamburg zwischen den Jahren 2017 und 2022, aufgeteilt nach Inhaltsbereichen [Zu16]. Da nicht jedes Fach in jedem Jahr angeboten wird, können einzelne Linien auch Lücken aufweisen.

Insgesamt lässt sich feststellen, dass die Bestehensquoten und Notenspiegel der Universität Hamburg **keinen eindeutigen Hinweis auf ein Problemfach** geben. Es wäre schön, wenn hier mehr Daten vorhanden wären, um Ausreißer durch die Eigenheiten einzelner Hochschulen auszuschließen. Zwar ist die Universität Hamburg eine sehr große Universität mit einem großen Fachbereich Informatik[4], allerdings lässt sich (auch unter Berücksichtigung unterschiedlicher Lehrpläne in weiterführenden Schulen der unterschiedlichen Bundesländer) zurecht die Frage stellen, ob die Universität Hamburg bundesweit repräsentativ ist. Zudem besteht hier das Problem, dass es durchaus den Wunsch geben könnte, möglichst viele Student*innen bestehen zu lassen und dies in die Bewertung mit einfließt, zum Beispiel, indem die Hürde für das Bestehen nach unten korrigiert wird. Dadurch könnte in den Daten, insbesondere in den Problemfächern, ein systematischer Fehler entstehen.

3 Betrachtung einzelner Fächer

Nachdem bei den Studienabbruchgründen (Abschnitt 2.1) und der Wahrnehmung von Dozent*innen und Student*innen (Abschnitt 2.2) mögliche Problemfächer vor allem in mathematisch-theoretischen und programmiernahen Fächern gefunden wurden, sollen diese Fächer hier noch einmal gesondert betrachtet werden. Dass bei den Noten selber keine Problemfächer (Abschnitt 2.3) gefunden wurden schließt nicht aus, dass in diesen beiden Bereichen nicht doch Probleme vorhanden sind.

3.1 Spezialfall: Mathematisch-Theoretische Fächer

Tedre und Sutinen [TS08] argumentieren, dass moderne Informatik sich auf drei Säulen aufbaut: Ingenieurwissenschaften, Naturwissenschaften und Mathematik („*mathematical*“, „*scientific*“ und „*engineering traditions*“). Entsprechend gibt es in der Informatik also eine technische, eine wissenschaftliche und eine mathematisch-theoretische Komponente.

4 Nach Eigenauskunft des Fachbereichs Informatik zuletzt ca. 350 Studienanfänger*innen im Bachelor bei ca. 2500 Student*innen insgesamt; https://www.inf.uni-hamburg.de/home/about/facts.html, abgerufen 2023-05-13.

Tatsächlich ist Mathematik ein fundamentaler Bestandteil auch angewandter Informatik: Programmierung basiert auf Funktionen, Konstanten und Variablen; Koordinaten in Videospielen arbeiten auf Basis linearer Algebra; die verschiedenen Fourier-Operationen in der Signalverarbeitung sind grundlegend analytische Verfahren; Netzwerktechnik macht sich Erkenntnisse der Graphentheorie zunutze – um nur einige Beispiele zu nennen. Dies zieht noch nicht in Betracht, dass die Algorithmik sowie die formalen Beweistechniken – beispielsweise über Mengeninklusion oder Induktion – kernmathematisch sind. Beaubouef [Be02] gibt einen guten Einblick darin, wo Mathematik in die Informatik eingewoben ist, und nennt dabei auch Beispiele, die hier schon genannt wurden. Letztendlich postuliert Beaubouef [Be02], dass auf einer gewissen Abstraktionsebene jeder Aspekt von Computern auf Zahlen, Arithmetik und Logik zurückzuführen sei, und schließt mit: „*Is it any wonder then, that computer science, the study of how to use a machine based on numbers to solve real-world problems requires a good understanding of math?*“ ([Be02, S. 59]).

Baldwin, Walker und Henderson [BWH13] haben die mathematischen Anforderungen einer Reihe von Informatik-Studienangeboten ermittelt und festgestellt, dass fast alle betrachteten Angebote diskrete Mathematik und Analysis voraussetzen, mit einem geringeren Anteil auch Stochastik und Statistik. Die von [BWH13] betrachteten Studienangebote bewegen sich zwischen einem und acht Mathematik-Kursen im Rahmen des Studiums. Dabei stellen sie fest, dass zwar viel Mathematik gelehrt wird, aber diese sich geringer (direkter) Anwendung im Studium erfreut: “*[. . .] while most such curricula include appropriate mathematics, they often also include much mathematics that is not strongly connected to computing, and while they teach some applications of math to computing, they often overlook others.*” [BWH13, S. 79]. Es lässt sich also eine Asymmetrie feststellen zwischen der Mathematik, die man im Studium aktiv verwendet, und der Mathematik, mit der man in mathematischen Modulen konfrontiert wird. Jerkins et al. [Je13] postulieren, dass Student*innen Mathematik als gesonderte Hürde auf dem Weg zum Informatik-Abschluss wahrnehmen würden (“*[. . .] students perceived the required mathematics coursework as simply a hoop to jump through to complete a CS degree.*”, [Je13]). Dies führe dazu, dass Student*innen die Wechselwirkungen zwischen Mathematik und Informatik nicht wahrnehmen würden, womit sie schlussendlich weniger Erfolg als Informatiker*innen erzielten.

Whalley, Petersen und Denny [WPD20] zeigen, dass es im Kontext von mathematisch-theoretischen Fächern in der Informatik zwei Gruppen von

Student*innen gibt: jene, die akademisch orientiert sind und den Mehrwert formaler Methoden für die Informatik erkennen, und jene, die berufsorientiert denken und theoretische Modelle als wenig zielführend erachten. Die Vermutung liegt nahe, dass die Gruppe, die ohnehin schon eine positive Disposition zu mathematisch-theoretischen Inhalten zeigt, auch weniger Schwierigkeiten in verwandten Fächern wahrnimmt. Analog ist anzunehmen, dass diejenigen Student*innen, die Mathematik als „Ballast“ wahrnehmen, in den theoretisch orientierten Kursen im Mittel subjektiv mehr Schwierigkeiten haben.

Interessant in dem Kontext ist eine Studie von Alpár et al. [Al22], die sich mit der Einstellung von Informatikstudent*innen zu Mathematik auseinandersetzt. Hier konnten die Autoren fünf Aspekte identifizieren, die sich auf den Eindruck von Student*innen auf Mathematik auswirken:

1. Vorherige Erfahrungen mit Mathematik
2. Soziales Umfeld
3. Selbstwirksamkeitserwartung
4. Angst vor Mathematik (*Math Anxiety*)
5. Motivationsquellen

Insbesondere diese Angst vor Mathematik (*Math Anxiety*) soll hier getrennt betrachtet werden, da es auch bei Alpár et al. von Teilnehmer*innen der Studie explizit als „*bottleneck*“ [Al22, S. 222] genannt wurde. Es konnte gezeigt werden, dass, wenn Student*innen Angst vor Mathematik haben, dies sich unter anderem in schlechterer Leistung und Erinnerung auswirkt [AK01]. Zudem scheint es eine negative Korrelation zwischen einer Angst vor Mathematik bzw. der Einstellung dazu und *mathematischem Denken* zu geben [KTB10]. Es liegt die Vermutung nahe, dass sich das Problem der *Math Anxiety* auch auf die theorielastigen Fächer übertragen lässt, da diese oftmals mathematiknah sind. Dies könnte zu einem Teil erklären, warum es sich hier um Problemfächer handelt.

Dem gegenüber stehen die Ergebnisse von Frede und Knobelsdorf [FK18], die ermitteln, dass die Leistung von Student*innen in der theoretischen Informatik weniger von Motivation, sondern mehr von Mängeln im grundlegenden Verständnis theoretischer Modelle und formaler Techniken abhängt. Zum Beispiel konnten Knobelsdorf und Frede [KF16] in einer Studie zeigen, dass es Student*innen in Theoriefächern vermutlich an Fähigkeiten zum Lösen der Aufgaben fehlt, insbesondere im Bereich der Mathematik und des Verstehens der mathematischen Symbole.

Insgesamt lässt sich feststellen, dass **die mathematisch-theoretischen Fächer durchaus als Problemfach angesehen werden können**. Entscheidender Faktor, ob die Student*innen hier Probleme haben, ist vor allem die Einstellung zur Mathematik und im extremen Fall eine Angst vor Mathematik (*Math Anxiety*).

3.2 Spezialfall: Programmiernahe Fächer

Die Diskussion, ob das Lernen von Programmieren schwierig (vgl. [RRR03]) oder nicht schwierig (vgl. [Lu16]) ist, wird schon länger geführt. An dieser Stelle kann daher nur ein kleiner Einblick in die Diskussion gegeben werden.

Betrachten wir zunächst die Durchfallquote der programmiernahen Fächer. Es gab in den letzten 20 Jahren einige Studien, die sich mit der Durchfall- und Abbruchquote in grundlegenden Programmierkursen befasst haben. Bennedsen und Caspersen [BC07] stellen fest, dass vor 2007 anekdotisch akzeptiert wurde, dass Programmierkurse schwer seien, konnten aber mit einer nach eigenen Angaben nicht repräsentativen Erhebung über Fachpersonen aus dem Kontext der *Computer Science Education (CSE)* keine harte Evidenz finden. Im Gegenteil berichten sie, dass im Schnitt etwa nur ein Drittel der Student*innen die einführenden Programmierkurse nicht besteht[5] und bezeichnen diese Quote als keine außergewöhnlich hohe Zahl (*„not an especially high percentage“*, [BC07, S. 35]) im Vergleich dazu, dass nur etwa ein Viertel der Studienanfänger tatsächlich auch den Abschluss erreichen würden.

Diese Studie wurde 2014 wieder aufgegriffen von Watson und Li [WL14], die eine größere Vergleichsstudie durchgeführt haben[6]. Sie berichten von einer durchschnittlichen Bestehensquote von 67,7 %, mit einem 90 % Konfidenz-Intervall zwischen 65,3 % und 70,1 %, und folgern daraus, dass ihre Ergebnisse nahezu perfekt zu denen von [BC07] passen würden und dass dadurch eine Bestehensquote von etwa zwei Dritteln möglicherweise ein realitätsnaher Wert sei. Watson und Li [WL14] stellen jedoch ihre Repräsentativität infrage, da sie lediglich die Ergebnisse von 161 Kursen über 15 Länder vergleichen.

5 Dies scheint unabhängig davon zu gelten, welches Programmierparadigma verfolgt wurde.

6 Es sollte der Vollständigkeit halber genannt werden, dass Kritik an der Methodologie und den daraus resultierenden Ergebnissen dieser Studie existiert: Guzdial, M. A Biased Attempt at Measuring Failure Rates in Introductory Programming, *Computing Education Research Blog*, September 30, 2014; https://computinged.wordpress.com/2014/09/30/a-biased-attempt-at-measuring-failure-rates-in-introductory-programming, abgerufen am: 02.12.2025.

Bennedsen und Caspersen [BC19] greifen ihre Studie später selber wieder auf und stellen – mit einer größeren Stichprobe – fest, dass die Bestehensquote mit 72,6 % in den zwölf Jahren seit ihrer ersten Studie besser geworden sei. Sie bleiben bei dem Ergebnis, dass die Durchfallquote nicht außergewöhnlich hoch sei.

Eine weitere Perspektive bringen Simon et al. [Si19] ein, die sich sowohl auf die beiden Studien von Bennedsen und Caspersen [BC07; BC19] als auch auf Watson und Li [WL14] stützen. Simon et al. [Si19] ermitteln eine Bestehensquote von etwa 75 %, etwas höher als kurz zuvor noch Bennedsen und Caspersen [BC19] bei noch größerer Stichprobe[7]. Auch die Daten der Universität Hamburg, wie sie in Abschnitt 2.3 präsentiert werden, liegen im Schnitt bei ungefähr 75 %. Dies deutet darauf hin, dass diese Daten repräsentativer sind als zunächst noch angenommen. Simon et al. [Si19] gehen allerdings noch einen Schritt weiter und vergleichen die Bestehensquoten von grundlegenden Programmierkursen mit anderen Einführungskursen im gesamten MINT-Bereich. In dem Kontext stellen sie fest, dass die Bestehensquoten einführender Programmierkurse sich nicht groß vom Mittel anderer einführender MINT-Kurse unterscheiden.

Zusätzlich zur Betrachtung der Fächer auf inhaltlicher Ebene sollte erwähnt werden, dass es viele weitere Faktoren gibt, die einen Einfluss auf den Lernerfolg haben, insbesondere das Vorwissen der Student*innen, wie zum Beispiel logisches Denken [DT22], Rechnerstrukturen [Lo22] sowie die Methodik des Programmierunterrichts (siehe [BF22; Zh22; MMD20; Lu16]).

Interessant ist auch der Zusammenhang zwischen Mathematik und Programmierung. Hierdurch entsteht ein direkter Bezug zwischen den bereits genannten mathematisch-theoretischen Fächern (Abschnitt 3.1) und den programmiernahen Fächern. So werfen Attallah, Ilagure und Chang [AIC18] einen Blick in die Literatur, um zu ermitteln, welche Zusammenhänge es zwischen mathematischen und programmatischen Fähigkeiten von Student*innen existieren. Sie verweisen beispielsweise auf Owolabi, Olanipekun und Iwerima [OOI14], die u. a. die Angst vor Mathematik, die Angst vor Computern wie auch die Angst vor Programmierung als Prädiktoren für Erfolge im Programmieren mit Basic untersuchen – und dabei zu dem Schluss kommen, dass hauptsächlich Mathematik-Kompetenz einen signifikanten Effekt als Prädiktor nachweisen kann. Auch auf Coetzee [Co16] wird verwiesen; sie konnte in ihrer Arbeit durch verschiedene Methoden einen starken Zusammenhang zwischen mathematischem Denken und den Programmierfähigkeiten bei Studienanfängern im ersten Jahr feststellen.

7 Was die Frage aufwirft, ob die Bestehensquote auch mit der Stichprobengröße korreliert und nicht nur über die Zeit besser geworden ist.

Insgesamt lässt sich feststellen, dass sich zumindest auf Basis der Bestehensquoten keine empirische Evidenz finden lässt, dass Programmieren sonderlich schwerer sei als andere Fächer, da die Bestehensquoten von Programmierkursen in vergleichbaren Bereichen wie andere MINT-Fächer liegen. Indizien deuten höchstens darauf hin, dass die Lehr-Lern-Methodik nicht immer dem Kursinhalt angemessen ist; in den Worten von Luxton-Reilly [Lu16, S. 288]: "*Learning to program is easy – all we need to do is collectively shift our view, and teach to achievable outcomes.*" Demnach lässt sich also sagen, dass **programmiernahe Fächer tendenziell keine Problemfächer** sind.

4 Zusammenfassung & Diskussion

Wenn man alle oben genannten Aspekte zusammenfasst, so lässt sich feststellen, dass es zwei potentielle Kandidaten für Problemfächer gibt: mathematisch-theoretische Fächer sowie programmiernahe Fächer. Bei einem Blick in relevante Literatur kann man dabei sehen, dass die **mathematisch-theoretischen Fächer als besonders schwierig im Informatikstudium angesehen werden können**. Wichtigster Faktor hier scheint die Einstellung der Student*innen zur Mathematik zu sein. Daher sollte sowohl in zukünftiger Forschung als auch in der zukünftigen Gestaltung von Studiengängen in der Informatik diesem Bereich eine erhöhte Aufmerksamkeit zukommen. Bei programmiernahen Fächern konnten in der Literatur dagegen Hinweise gefunden werden, dass es sich nicht um Problemfächer handelt. Dabei sei der Vollständigkeit halber nochmal erwähnt, dass es einen Zusammenhang zwischen der Mathematik und dem Programmieren zu geben scheint (siehe auch [OOI14; AIC18; Co16]), sodass sich die Probleme in den mathematisch-theoretischen Fächern auch auf die programmiernahen Fächer auswirken könnten.

Diese Ergebnisse passen auch zu den Eindrücken von Teilnehmer*innen auf der *Hochschuldidaktik der Informatik HDI 2023*: Bei einer Präsentation der letzten Version dieses Beitrages [SKL23] wurden vorab die im Raum befindlichen Personen befragt, welche Fächer sie aus ihrer Erfahrung für Problemfächer halten. Auch hier wurden vor allem Mathematik und theorienahe Fächer oft genannt – andere Fachrichtungen waren (falls erwähnt) Einzelnennungen. Dementsprechend passen die Beobachtungen der Expert*innen auf der HDI zu den Beobachtungen in dieser Veröffentlichung.

Einige weitere Beobachtungen von den Teilnehmer*innen auf der HDI sollen hier noch kurz vorgestellt werden:

- Die Dozent*in selber hat einen großen Einfluss darauf, wie gut Studierende lernen. Hier wurde von den Teilnehmer*innen die Frage gestellt, ob nicht bestimmte Fächer Dozenten anziehen, die eher an Forschung als an Lehre interessiert sind. Dass Forschung in der Hochschule höher gestellt wird als Lehre ist ein bekanntes Problem (siehe z. B. [Pa08]). Genauso kann es sein, dass in bestimmten Fächern schwere Prüfungen gestellt werden, die Fächer selber aber nicht besonders schwer sind.
- Der Informatikunterricht ist in Deutschland noch sehr unterschiedlich verteilt [SHF21][8], während Mathematik in allen Schulen Pflicht ist. Dies könnte eine Auswirkung haben; so werden Mathekenntnisse eher noch erwartet als Programmierkenntnisse [SK22; Be23] (auch wenn beides nicht von einer Mehrheit der Dozent*innen vorausgesetzt wird).
- Eine Frage der Teilnehmer*innen war es zu prüfen, inwieweit der Aufwand, den Studierende in einen Kurs investiert haben, einen Einfluss hat. Einen ersten, nicht vollständigen Einblick bieten hierzu Baucks und Wiskott [BW23], die in ihrem Modell zeigen konnten, dass die Arbeitslast eines Semesters einen vernachlässigbaren Effekt auf die Noten hat. Hätte der investierte Arbeitsaufwand einen starken Effekt, so würde man erwarten, dass in einem Semester mit wenig Last die Noten besser wären. Mehr Forschung hier wäre sinnvoll.
- Offen ist auch die Frage, welche gemeinsamen Elemente die mathematisch-theoretischen Fächer haben (z. B. Rechenvorschrift formulieren, Eingabe-Verarbeitung-Ausgabe, Angst vor Mathematik), die den Fächern ihre Schwierigkeit geben. Wenn diese identifiziert werden können, dann ließen sich diese didaktisch adressieren.

In dieser Studie wird eine starke Vereinfachung vorgenommen, da sie sich auf den Einfluss einzelner Fächer konzentriert. Der Studienabbruch kann aber durchaus als ein Geschehen betrachtet werden, in das viele Faktoren einfließen (vgl. [He17, S. 11–16]). So können zum Beispiel die Fähigkeiten der Dozent*innen, die persönlichen Interessen und Stärken von Student*innen, die

8 Es gibt Bestrebungen, Informatik flächendeckend als Pflichtfach zu etablieren; so empfiehlt zum Beispiel die Ständige Wissenschaftliche Kommission der KMK eine Einführung ab Sekundarstufe I mit Schuljahr 2024/25, vgl. [SWK22].

persönliche Situation von Student*innen, die organisatorischen und rechtlichen Rahmenbedingungen wie Studienpläne und Prüfungsordnungen oder die Abhängigkeit zwischen verschiedenen Fächern eine Rolle spielen. Genauso werden unter Umständen die Bedürfnisse einzelner Gruppen vernachlässigt, wie z. B. von Student*innen mit Migrationshintergrund, von Student*innen mit verschiedenen Bildungshintergründen oder geschlechterspezifische Bedürfnisse. Trotzdem kann der Blick auf einzelne Fächer sinnvoll sein. Wenn die hier gefundenen Hinweise stimmen und theorielastige Fächer tatsächlich problematisch sind, so lohnt es sich, sowohl die Suche nach den Gründen hierfür zu intensivieren als auch neue didaktische und pädagogische Konzepte für diese Fächer zu entwickeln.

Es kann außerdem einen möglichen Einfluss auf den Lernerfolg haben, an welcher Stelle im Curriculum ein Kurs steht. So kritisieren Logozar et al. [Lo22], dass der von ihnen unterrichtete Kurs aufgrund der hohen Dichte ihres Curriculums zu weit nach vorne gerutscht sei und somit eigentlich vorbereitende Inhalte erst parallel oder gar in einem späteren Semester vermittelt werden. Als mildere Mittel beschreiben sie Addenda zu den Lehrmaterialien und das Angebot eines Aufbaukurses. Hier lässt sich festhalten, dass bei der Gestaltung und Verschiebung von Curricula die inhaltlichen Zusammenhänge und (implizite und explizite) Modulabhängigkeiten im Blick behalten werden sollten. Im Gegensatz dazu zeigen Baucks und Wiskott [BW23], dass in ihrem Modell nur schwache Wechselwirkungen zwischen den meisten Kursen bestehen.

Genauso besteht die Möglichkeit, dass einzelne Institutionen ein Fach bewusst zu einem Problemfach erheben (z. B. durch hohe Anforderungen), um eine Hürde für Student*innen aufzubauen. Ziel hierbei könnte die Reduzierung von Student*innenzahlen oder das frühzeitige Herausprüfen von vermeintlich zu schlechten Student*innen sein. Ob und inwieweit ein solches Vorgehen sinnvoll ist, soll jedoch nicht in diesem Beitrag betrachtet werden.

Leider ist die aktuelle Datenlage zu dieser Fragestellung nicht zufriedenstellend. Daten liegen entweder vereinzelt vor (z. B. bei den Bestehensquoten / Notenspiegeln), erlauben nur indirekte Schlussfolgerungen (z. B. bezüglich Studienabbruchgründen) oder sind durch bestehende Vorurteile beeinflusst (z. B. bei den Wahrnehmungen von Dozent*innen und Student*innen). Auch eine gezielte Studie aufzusetzen, ist nicht einfach möglich – zum einen handelt es sich hierbei um eine Situation mit vielen Variablen, zum anderen wird eine Befragung immer die bereits genannten Vorurteile beibehalten. Daher scheint

hier derzeit die beste Möglichkeit eine Annäherung an die Frage, wie es in diesem Beitrag versucht wurde.

Wünschenswert wäre eine Kooperation von mehr Hochschulen, um mehr Noten- und Bestehensquoten zum Abgleich verfügbar zu haben. Bisher bleiben diese Daten oftmals unter der Kontrolle einer einzelnen Hochschule, wodurch Analysen dieser Daten über Institutionsgrenzen hinweg nicht möglich sind.

Acknowledgment

Ein besonderer Dank geht an FragDenStaat (https://fragdenstaat.de/) für die Bereitstellung einer Plattform für Informationsfreiheitsanfragen. Die Anfrage für die in dieser Arbeit verwendeten Daten findet sich unter https://fragdenstaat.de/a/243870.

Literatur

[AIC18] Attallah, Belsam, Ilagure, Zakea und Chang, Yun-Ke: The Impact of Competencies in Mathematics and Beyond on Learning Computer Programming in Higher Education. In: 2018 Fifth HCT Information Technology Trends (ITT). S. 77–81, 2018.

[AK01] Ashcraft, Mark H. und Kirk, Elizabeth P.: The relationships among working memory, math anxiety, and performance. Journal of Experimental Psychology: General 130 (2), S. 224–237, 2001.

[Al22] Alpár, Greg, Yeni, Sabiha, Aivaloglou, Efthimia und Hermans, Felienne: Can Math Be a Bottleneck? Exploring the Mathematics Perceptions of Computer Science Students. In: 2022 IEEE Global Engineering Education Conference (EDUCON). S. 217–225, 2022.

[BC07] Bennedsen, Jens und Caspersen, Michael E.: Failure Rates in Introductory Programming. SIGCSE Bull. 39 (2), S. 32–36, 2007.

[BC19] Bennedsen, Jens und Caspersen, Michael E.: Failure Rates in Introductory Programming: 12 Years Later. ACM Inroads 10 (2), S. 30–36, 2019.

[Be02] Beaubouef, Theresa: Why Computer Science Students Need Math. SIGCSE Bull. 34 (4), S. 57–59, 2002.

[Be10] Bethlehem, Jelke: Selection Bias in Web Surveys. International Statistical Review 78 (2), S. 161–188, 2010.

[Be21] Behr, Andreas, Giese, Marco, Teguim Kamdjou, Herve D. und Theune, Katja: Motives for dropping out from higher education—An analysis of bachelor's degree students in Germany. European Journal of Education 56 (2), S. 325–343, 2021.

[Be23] Bender, Esther, Barbas, Helena, Hamann, Fabian, Soll, Marcus und Sitzmann, Daniel: Fähigkeiten und Kenntnisse bei Studienanfänger*innen in der Informatik: Was erwarten die Dozent*innen? In (Desel, Jörg, Opel, Simone und Siegeris, Juliane, Hrsg.): Hochschuldidaktik Informatik HDI 2021 (Commentarii informaticae didacticae). Bd. 13, S. 279–299, 2023.

[BF22] Berssanette, João Henrique und de Francisco, Antonio Carlos: Cognitive Load Theory in the Context of Teaching and Learning Computer Programming: A Systematic Literature Review. IEEE Transactions on Education 65 (3), S. 440–449, 2022.

[Bö21] Böttcher, Axel, Thurner, Veronika, Häfner, Tanja und Ottinger, Sarah: Adaptierung von Beratungsangeboten auf der Basis von Erkenntnissen aus der Analyse von Studienverlaufsdaten. In: 9. Fachtagung Hochschuldidaktik Informatik (HDI) 2021. S. 57–64, 2021.

[BW23] Baucks, Frederik und Wiskott, Laurenz: Mitigating Biases using an Additive Grade Point Model: Towards Trustworthy Curriculum Analytics Measures. In: 21. Fachtagung Bildungstechnologien (DELFI). Gesellschaft für Informatik e.V., Bonn, S. 41–52, 2023.

[BWH13] Baldwin, Douglas, Walker, Henry M. und Henderson, Peter B.: The Roles of Mathematics in Computer Science. ACM Inroads 4 (4), S. 74–80, 2013.

[Co16] Coetzee, Carla: Mathematical thinking skills needed by first year programming students, Diss., Pretoria: University of Pretoria, 2016, http://hdl.handle.net/2263/60991, abgerufen am: 27. 10. 2025.

[DT22] Daungcharone, Kannika und Thongkoo, Krittawaya: The Effects of Thinking Process Model Technique on Logical Thinking Skills Influencing Programming Achievement. In: 2022 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON). S. 134–138, 2022.

[FK18] Frede, Christiane und Knobelsdorf, Maria: Exploring how Students Perform in a Theory of Computation Course using Final Exam and Homework Assignments Data. In (Malmi, Lauri, Korhonen, Ari, McCartney, Robert und Petersen, Andrew, Hrsg.): Proceedings of the 2018 ACM Conference on International Computing Education Research. ICER 2018, Espoo, Finland 13.–15. August 2018. ACM, S. 241–249, 2018.

[He10] Heublein, U., Hutzsch, C., Schreiber, J., Sommer, D. und Besuch, G.: Ursachen des Studienabbruchs in Bachelor- und in herkömmlichen Studiengängen: Ergebnisse einer bundesweiten Befragung von Exmatrikulierten des Studienjahres 2007/08. 2010.

[He17] Heublein, Ulrich, Ebert, Julia, Hutzsch, Christopher, Isleib, Sören, König, Richard, Richter, Johanna und Woisch, Andreas: Zwischen Studienerwartungen und Studienwirklichkeit. Ursachen des Studienabbruchs, beruflicher Verbleib der Studienabbrecherinnen und Studienabbrecher und Entwicklung der Studienabbruchquote an deutschen Hochschulen. Hannover, 2017.

[HSW06] Holdt, U. v., Schneider, H. und Wagner, B.: Analyse von Studienverläufen und Studienabbrüchen in den Bachelorstudiengängen Informatik an der Leibniz Universität Hannover. In (Forbrig, Peter, Siegel, Günter und Schneider, Markus, Hrsg.): HDI 2006: Hochschuldidaktik der Informatik – Organisation, Curricula, Erfahrungen. Gesellschaft für Informatik e. V., Bonn, S. 115–126, 2006.

[Je13] Jerkins, James A., Stenger, Cynthia L., Stovall, Jessica und Jenkins, Janet T.: Establishing the Impact of a Computer Science/Mathematics Anti-Symbiotic Stereotype in CS Students. Journal of Computing Sciences in Colleges 28 (5), S. 47–53, 2013, https://dl.acm.org/doi/abs/10.5555/2458569.2458578, abgerufen am: 27. 10. 2025.

[Jo05] Jonkmann, Kathrin: Studienabbruch, Studiendauer und Studienerleben - Analyse der Studierendenumfrage des Instituts für Informatik der Humboldt-Universität zu Berlin. 2005.

[Jo99] Joint declaration of the European Ministers of Education: The Bologna Declaration of 19 June 1999. 1999, https://ehea.info/Upload/document/ministerial_declarations/1999_Bologna_Declaration_English_553028.pdf, abgerufen am: 03. 08. 2023.

[KB00] King, Maryon F. und Bruner, Gordon C.: Social desirability bias: A neglected aspect of validity testing. Psychology & Marketing 17 (2), S. 79–103, 2000.

[KF16] Knobelsdorf, Maria und Frede, Christiane: Analyzing Student Practices in Theory of Computation in Light of Distributed Cognition Theory. In: Proceedings of the 2016 ACM Conference on International Computing Education Research. ICER '16, Melbourne, VIC, Australia. Association for Computing Machinery, New York, NY, USA, S. 73–81, 2016.

[KTB10] Kargar, Maryam, Tarmizi, Rohani Ahmad und Bayat, Sahar: Relationship between Mathematical Thinking, Mathematics Anxiety and Mathematics Attitudes among University Students. Procedia – Social and Behavioral Sciences 8, International Conference on Mathematics Education Research 2010 (ICMER 2010), S. 537–542, 2010.

[Lo22] Logozar, Robert, Horvatic, Miroslav, Sumiga, Ivan und Mikac, Matija: Challenges in Teaching Assembly Language Programming – Desired Prerequisites vs. Students' Initial Knowledge. In: 2022 IEEE Global Engineering Education Conference (EDUCON). S. 1689–1698, 2022.

[Lu16] Luxton-Reilly, Andrew: Learning to Program is Easy. In: Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education. ITiCSE '16, Arequipa, Peru. Association for Computing Machinery, New York, NY, USA, S. 284–289, 2016.

[MMD20] Margulieux, Lauren E., Morrison, Briana B. und Decker, Adrienne: Reducing withdrawal and failure rates in introductory programming with subgoal labeled worked examples. International Journal of STEM Education 7 (1), 2020.

[NHD19] Neugebauer, Martin, Heublein, Ulrich und Daniel, Annabell: Studienabbruch in Deutschland: Ausmaß, Ursachen, Folgen, Präventionsmöglichkeiten. Zeitschrift für Erziehungswissenschaft 22 (5), S. 1025–1046, 2019.

[Ni98] Nickerson, Raymond S.: Confirmation Bias: A Ubiquitous Phenomenon in Many Guises. Review of General Psychology 2 (2), S. 175–220, 1998.

[OOI14] Owolabi, J., Olanipekun, P. und Iwerima, J.: Mathematics Ability and Anxiety, Computer and Programming Anxieties, Age and Gender as Determinants of Achievement in Basic Programming. GSTF Journal on Computing (JoC) 3 (4), S. 47, 2014.

[Pa08] Parker, Jonathan: Comparing Research and Teaching in University Promotion Criteria. Higher Education Quarterly 62 (3), S. 237–251, 2008.

[RRR03] Robins, Anthony, Rountree, Janet und Rountree, Nathan: Learning and Teaching Programming: A Review and Discussion. Computer Science Education 13 (2), S. 137–172, 2003.

[Sa21] Salguero, Adrian, Griswold, William G., Alvarado, Christine und Porter, Leo: Understanding Sources of Student Struggle in Early Computer Science Courses. In: Proceedings of the 17th ACM Conference on International Computing Education Research. ICER 2021, Virtual Event, USA. Association for Computing Machinery, New York, NY, USA, S. 319–333, 2021.

[SHF21] Schwarz, Richard, Hellmig, Lutz und Friedrich, Steffen: Informatikunterricht in Deutschland – eine Übersicht. Informatik Spektrum 44 (2), S. 95–103, 2021.

[Si19] Simon et al.: Pass Rates in Introductory Programming and in Other STEM Disciplines. In: Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education. ITiCSE-WGR '19, Aberdeen, Scotland Uk. Association for Computing Machinery, New York, NY, USA, S. 53–71, 2019.

[SK22] Soll, Marcus und Kobras, Louis: What Were We Expecting? Analysing Expectations of German University Teachers of Study Beginners in Computer Science as Experienced by Students. In: 2022 IEEE German Education Conference (GeCon). S. 1–6, 2022.

[SKL23] Soll, Marcus, Kobras, Louis und Lagod, Christian: Gibt es ein Problemfach im Informatikstudium? In: 10. Fachtagung Hochschuldidaktik Informatik (HDI) 2023. S. 89–103, 2023.

[SWK22] Digitalisierung im Bildungssystem: Handlungsempfehlungen von der Kita bis zur Hochschule. Gutachten der Ständigen Wissenschaftlichen Kommission der Kultusministerkonferenz (SWK), Bonn: Ständige Wissenschaftliche Kommission der Kultusministerkonferenz (SWK), 2022.

[TS08] Tedre, Matti und Sutinen, Erkki: Three traditions of computing: what educators should know. Computer Science Education 18 (3), S. 153–170, 2008.

[Ts19] Tsarouha, Elena: Aussagekraft und Vergleichbarkeit von Noten. In: Prüfungspraktiken an deutschen Hochschulen: Eine empirische Studie zu systematischen Einflussgrößen auf die Notengebung in Abschlussprüfungen. Springer Fachmedien Wiesbaden, Wiesbaden, S. 39–46, 2019.

[WL14] Watson, Christopher und Li, Frederick W.B.: Failure Rates in Introductory Programming Revisited. In: Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education. ITiCSE '14, Uppsala, Sweden. Association for Computing Machinery, New York, NY, USA, S. 39–44, 2014.

[WPD20] Whalley, Jacqueline, Petersen, Andrew und Denny, Paul: Mathematics, Computer Science and Career Inclinations — A Multi-Institutional Exploration. In: Proceedings of the 20th Koli Calling International Conference on Computing Education Research. Koli Calling '20, Koli, Finland. Association for Computing Machinery, New York, NY, USA, 2020.

[Zh22] Zhao, Dan, Muntean, Cristina Hava, Chis, Adriana E., Rozinaj, Gregor und Muntean, Gabriel-Miro: Game-Based Learning: Enhancing Student Experience, Knowledge Gain, and Usability in Higher Education Programming Courses. IEEE Transactions on Education 65 (4), S. 502–513, 2022.

[Zu16] Zukunft, Olaf: Empfehlungen für Bachelor- und Masterprogramme im Studienfach Informatik an Hochschulen (Juli 2016), Bonn: Gesellschaft für Informatik e. V., 2016.