How Do Students Talk About Intelligence? An Investigation of Motivation, Self-efficacy, and Mindsets in Computer Science

Jamie Gorson
Northwestern University
Evanston, IL
jgorson@u.northwestern.edu

Eleanor O'Rourke
Northwestern University
Evanston, IL
eorourke@northwestern.edu

ABSTRACT

Undergraduate programs in computer science (CS) face high dropout rates, and many students struggle while learning to program. Studies show that perceived programming ability is a significant factor in students' decision to major in CS. Fortunately, psychology research shows that promoting the growth mindset, or the belief that intelligence grows with effort, can improve student persistence and performance. However, mindset interventions have been less successful in CS than in other domains. We conducted a small-scale interview study to explore how CS students talk about their intelligence, mindsets, and programming behaviors. We found that students' mindsets rarely aligned with definitions in the literature; some present mindsets that combine fixed and growth attributes, while others behave in ways that do not align with their mindsets. We also found that students frequently evaluate their self-efficacy by appraising their programming intelligence, using surprising criteria like typing speed and ease of debugging to measure ability. We conducted a survey study with 103 students to explore these self-assessment criteria further, and found that students use varying and conflicting criteria to evaluate intelligence in CS. We believe the criteria that students choose may interact with mindsets and impact their motivation and approach to programming, which could help explain the limited success of mindset interventions in CS.

CCS CONCEPTS

• **Social and professional topics → Computer science education**; **CS1**;

KEYWORDS

Growth Mindset; Self-efficacy; Motivation; Qualitative methods.

ACM Reference format:
Jamie Gorson and Eleanor O'Rourke. 2019. How Do Students Talk About Intelligence? An Investigation of Motivation, Self-efficacy, and Mindsets in Computer Science. *In International Computing Education Research Conference (ICER '19), August 12–14, 2019, Toronto, ON, Canada.* ACM, New York, NY, USA 9 pages. https://doi.org/10.1145/3291279.3339413

ICER '19, August 12–14, 2019, Toronto, ON, Canada

ACM ISBN 978-1-4503-6185-9/19/08…$15.00
https://doi.org/10.1145/3291279.3339413

1 INTRODUCTION

Software development is one of the fastest growing occupations in the United States [1], and the demand for skilled programmers is growing rapidly. However, many students struggle to learn computer science, and introductory courses often have high dropout rates [4, 5]. This problem is particularly pronounced for women and underrepresented minorities [17], who dropout of CS at a higher rate than their counterparts [8]. As a result, researchers are interested in understanding student motivation to inform the design of interventions that help reduce the dropout rate and increase diversity in CS.

Researchers have studied a number of factors that impact student motivation in CS, including self-efficacy [26, 27, 35, 40], emotional reactions to programming assignments [26, 28], attributions of successes and failures [43], sense of belonging [42], and self-regulation [29, 30]. Mindsets about intelligence are one factor that may have a particularly strong impact on retention and diversity in CS. Psychology research shows that students who believe intelligence is an inborn trait (fixed mindset) value proving their intelligence over learning, and are more likely to give up when challenged to avoid failure [11, 12, 14]. In contrast, students who believe that intelligence is malleable (growth mindset) value learning over performance, and tend to persist in the face of challenges [11, 12, 23]. Fortunately, mindset researchers have developed interventions that promote the growth mindset, improving student persistence and performance [2, 6, 44], and reducing the negative impacts of stereotype threat for women and underrepresented minorities [2, 20].

In computer science, the fixed mindset is particularly prevalent. Multiple studies have shown that students' mindsets become significantly more fixed during their first programming course [9, 18, 30]. Additionally, researchers found that CS students consider their self-assessments of ability when deciding to major or persist in CS, and interpret these assessments in different ways based on their mindset [27]. However, we still have a limited understanding of how mindsets are enacted in CS, and how to design mindset interventions for this context. For example, Kaijanaho et al. found that there was no correlation between mindset and performance in two advanced CS courses [25]. Furthermore, multiple studies have shown that traditional mindset interventions have little impact on CS students' mindsets and behaviors [9, 38]. In order to design effective interventions, we first need to understand how students think about intelligence in computer science, and how this impacts their mindsets, motivations, and approaches to programming.

In this paper, we contribute two studies that explore how novice university students define and measure intelligence in CS, and how this affects their perception of programming. Through in-depth qualitative interviews with nine students, the first study

explores how students talk about intelligence and mindsets in CS, and how they react to struggle and challenge in their CS classes. We found that only one participant's talk aligned with mindset theory; the other eight participants' talk either included both fixed and growth attributes or misaligned with their approaches to programming. We also noticed, during our interviews, that students often evaluated their self-efficacy by appraising their programming intelligence, using surprising criteria like typing speed and ease of debugging to measure ability. Our second study explores these self-assessment criteria in more depth through a survey of 103 students. We found significant variation in the criteria, showing that students define intelligence in CS in very different ways. We believe that self-assessment criteria may interact with mindsets and impact students' motivations and approaches to programming, which could help explain the limited success of mindset interventions in CS. These findings suggest a need for more research to understand the relationship between self-efficacy and mindsets in CS.

2 RELATED WORK

A number of studies have explored students' beliefs about intelligence and motivation in CS from a variety of theoretical perspectives. We first present research on mindset theory and then describe how the theory has been applied to the computer science domain. Finally, we discuss research that explores students' perceived ability in CS and their decision to pursue the major, which could relate to self-efficacy and mindset.

2.1 Mindset Theory

Research in psychology has demonstrated that students' beliefs about the malleability of intelligence can have a strong impact on their motivation, reaction to challenge, and academic performance [6, 20]. Students with a *growth mindset* believe that intelligence is malleable, and can grow through effort and practice [11, 12, 14]. Students with the growth mindset value learning over performance, and are more likely to persist when challenged [6, 32]. Students with a *fixed mindset* believe that intelligence is an unchangeable attribute, and that there is a limit to each person's potential growth [11, 12, 14]. Students with the fixed mindset view challenges as tests of their intelligence, and see mistakes as evidence of low ability [6, 32]. For the remainder of the paper, we call the behaviors that studies show correlate with mindset *associated behaviors*.

Fortunately, studies show that students' mindsets, and subsequently these associated behaviors, can be changed through interventions [2, 6, 44]. Researchers have developed two types of successful interventions. The first involves directly teaching about the malleability of intelligence and the growth mindset through readings and discussions. For example, Aronson et al. taught participants about the growth mindset and then asked them to write to younger students to teach them about the malleability of intelligence, creating a 'saying is believing' effect [2]. The second type of intervention involves changing the type of praise given to students when they are successful. Students who are praised for their effort when they succeed (e.g. "you must have worked hard") are more likely to develop a growth mindset and react favorably to subsequent failure than students who are praised for their ability (e.g. "you must be smart") [21, 32, 34].

2.2 Mindsets in CS

While mindset interventions have been successful in many domains, they have not been shown to change students' mindsets or behaviors in CS [9, 38]. For example, Cutts et al. designed a direct teaching intervention in which tutors gave mindset lessons and messages to CS students [9]. While the intervention successfully changed students' mindsets on surveys, they did not find any change in course performance. Simon et al. replicated Aronson et al.'s successful intervention [2], in which they taught students about mindset theory and conducted a 'saying is believing' exercise, but found no significant effect on students' mindsets in the domain of programming [38]. Loksa et al. designed an intervention for CS youth camps that taught students problem-solving skills [30]. They found that students in a control group became more fixed mindset over the course of the camp, but encouragingly they saw no significant change in mindsets for the intervention group. Additionally, students in the intervention group were more likely to persist on problems before asking for help. However, it is still unclear what aspects of Loksa et al.'s intervention produced these effects, and whether they were a result of student mindsets, the problem-solving scaffolds, or student age.

After their intervention had such little impact on mindset survey responses, Simon et al. conducted exploratory research on the reliability of the mindset survey [38]. They prompted students to explain their answers to the mindset survey and found that the survey does not always accurately capture the way students describe their own mindsets. Additionally, some students expressed aspects of both mindsets. Our research uses qualitative methods to understand how students talk about intelligence and study the nuances of how mindsets are enacted in the domain of CS. We provide new evidence showing that students' mindsets do not always match what we would expect based on mindset theory, which could help explain some of these surprising findings in CS.

2.3 Relationship Between Students' Perceived Ability and Persistence in CS

Researchers have also explored how students' evaluations of ability impact their emotions and persistence in CS through lenses other than mindset theory. Lewis et al. conducted an interview study that explores the factors that shape students' decision to major in CS [27]. They found that students' self-assessments of their programming ability impact their persistence, and that students judge their ability based on three criteria: perceived prior experience, speed, and grades. Our findings extend this work by studying how students assess performance while programming instead of in CS generally, revealing a larger and more specific set of self-assessment criteria that could help explain how students think about and define ability in CS.

Kinnunen and Simon found that students' expectations about programming also influence how they react to programming experiences [26]. They found that even when students have positive programming outcomes, they sometimes still have negative emotional reactions when their performance expectations are not met. Emotional reactions in programming are important to study because they can have a negative impact on learning outcomes [28]. While Kinnunen and Simon found that students have expectations

that influence their self-efficacy and motivation, they do not specify or study these expectations in depth. We build on this work by exploring expectations, which we call self-assessment criteria.

3 INTERVIEW STUDY

We designed our first study to explore how novice undergraduate students talk about intelligence in CS. Our goal was to study how this talk reflected students' (1) mindsets, (2) associated behaviors, like persistence and reaction to challenge, and (3) other motivational factors. To address these questions, we took a qualitative approach. In our study, students first worked on a challenging programming problem, then filled out a mindset survey, and finally responded to interview questions about their experiences in CS, perspectives on programming intelligence, and mindsets. The open-ended nature of our interviews and our inductive qualitative analysis allows us to broadly explore how students' perspectives on intelligence might influence their programming behaviors.

3.1 Participants and Setting

We recruited nine undergraduate students at a large private university who were enrolled in CS 1.5, a course designed for students who do not feel ready to move on to the second course in the CS sequence. We chose this demographic because the students have been exposed to programming, but still consider themselves to be novices. Additionally, many students in this course are still deciding whether or not to major in CS, and thus have thought about their ability to succeed in the field. Five (55%) of our participants were female, which is representative of the 61% in the course, and two (22%) of our participants had declared CS majors, which is representative of the 26% of students in the course who had declared. All students provided informed consent to participate and were compensated at a rate of $30 per hour.

3.2 Procedure

Our study procedure included three key tasks: a challenging programming problem, the mindset survey, and a clinical interview [19]. First, we asked participants to work on a challenging programming problem for twenty-five minutes. The goal of the programming task was to elicit interesting thoughts and feelings about intelligence and mindset that can be later discussed in the interview, since studies show that mindsets only impact behaviors when students are challenged [12]. Additionally, we wanted all participants to have a similar experience before the survey and clinical interview so they would be in a comparable emotional state.

Next, we asked participants to complete a mindset survey [11], so that we could compare our qualitative evaluation of their mindset to the canonical mindset measure. We adapted the traditional survey to ask about programming aptitude instead of general intelligence because studies show that domain-specific mindset surveys are more accurate [37]. We administered the survey after the programming problem, but before the interview, so that the conversation with the researcher would not impact the survey responses.

Finally, the first author conducted a clinical interview [19], during which the researcher asks participants a prepared set of questions and adds follow-up questions as relevant topics arise. First, we asked participants questions that elicited mindsets indirectly. For example, we started with questions about the programming task ("how well do you think you did on that problem?") to develop rapport. We gradually asked more general questions about their programming experiences ("tell me about a time when you struggled on a programming problem") and then about opinions on programming intelligence ("what do you think it takes to succeed in CS?"). At the very end of the interview, we asked about their mindset in CS directly to see if they self-identify with one of the mindsets without biasing their responses to the earlier questions.

3.3 Data Analysis

We analyzed the data by using a combination of deductive and inductive qualitative coding to create a theoretically informed codebook [31, 39, 41]. We developed initial deductive codes based on research that outlines the relationship between mindsets and behaviors. For example, the attribution literature shows that students who have a growth mindset are more likely to attribute their success to effort [14, 36], so we created an effort attribution code. We also created inductive codes for other types of talk that related to mindset, intelligence, and persistence. This open coding allowed us to identify emergent themes that we were not expecting based on the mindset literature. We iterated on our codes until their definitions were clear. Then, the two authors independently coded all of the data and discussed any discrepancies. There are 14 codes in the final codebook.

In the codebook, four pairs of codes capture cases where participants' talk exposes their beliefs about the malleability of programming intelligence, which we call *mindset talk*. For example, when a participant states that there is a limit to their potential growth in CS, we would label this as fixed mindset talk. Three additional pairs of codes capture cases where a participant either behaves or talks about behaving in a way that literature shows is associated with a mindset, which we call *associated behaviors*. For example, if a participant says that struggle is good because it results in learning, we would label this as an associated behavior of the growth mindset. The codebook is shown in Table 1.

One limitation of qualitatively coding interviews to study mindsets is that we must interpret students' statements and make judgements about their meaning. While prior research shows that people with certain mindsets tend to say certain things and behave in certain ways, we cannot definitively know an individual's mindset from their talk alone. However, this type of qualitative analysis allows us to gain a deeper understanding of how student beliefs are enacted in real contexts than we can achieve through Likert-scale surveys. We therefore believe this approach will provide important new insights into student mindsets about intelligence in CS.

3.4 Mindset Findings

To analyze how our participants' mindsets are enacted through their talk we counted the number of statements that were labeled with either mindset talk codes or associated behavior codes for each participant. Then, we calculated the percentage of mindset talk that was coded as growth and the percentage of associated behavior talk that was coded as growth. Based on the literature, we would expect that most students would have one consistent mindset and the corresponding associated behaviors. For example,

Growth Mindset Talk Codes	Fixed Mindset Talk Codes
Attributes an outcome to effort or learning	Attributes an outcome to their ability
States that growth in CS is possible with effort	States that there is a limit to their ability or potential for success in CS
States that peers are different based on controllable reasons	States that peers are different based on innate reasons
Self-identifies as growth mindset	Self-identifies as fixed mindset
Growth Mindset Associated Behavior Codes	**Fixed Mindset Associated Behavior Codes**
States that struggle, practice, or challenge is good	Doesn't value effort, struggle, practice, or challenge
Asks researcher for help on programming task	Asks researcher about performance on the programming task
Is motivated, persists, or seeks out a learning opportunity	Avoids a learning opportunity or programming activity

Table 1: Qualitative codes used to analyze our interviews, including codes that indicate either a fixed or growth mindset, and codes that indicate behaviors that are associated with either a fixed or growth mindset.

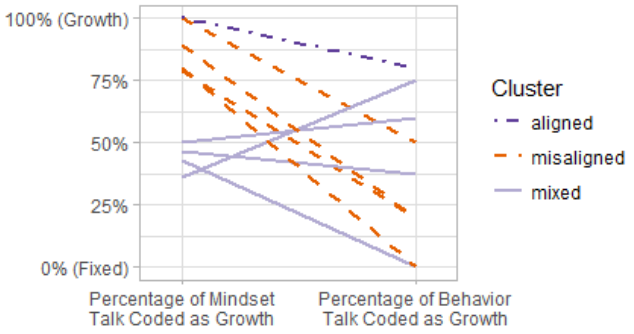

Figure 1: Graph showing the percentage of mindset talk that was coded as growth (left) and the percentage of associated behaviors that were coded as growth (right). Participants are grouped into three clusters: the first includes the participant whose talk *aligned* with mindset theory; the second includes participants with *misaligned* mindset talk and associated behaviors; the third includes *mixed* mindset participants.

if the majority of a student's mindset talk was growth, we would also expect most of their associated behavior talk to be growth.

We used the percentages of growth mindset talk and associated behavior talk to classify and cluster the participants. We classified participants as having a growth mindset if at least 75% of their mindset talk was labeled with growth codes, and fixed if 75% of their mindset talk was labeled fixed. We classified participants as having growth mindset associated behaviors if at least 75% of their associated behavior talk was labeled with growth codes, and fixed if 75% of their associated behavior talk was labeled fixed. To confirm that we agreed with the classifications, we read all of the statements coded as mindset talk or associated behaviors for each participant. Finally, we compared our classifications with the participants' self-identified mindsets and responses to the mindset survey.

Next, we identified clusters of participants by analyzing the relationship between their mindset talk and associated behavior classifications, as shown in Figure 1. The three clusters that we identified are: *aligned with mindset theory*, *misaligned mindset and associated behaviors*, and *mixed mindset*. We define each cluster in more detail in the sections that follow, and present one participant from each cluster as a case study.

3.4.1 Aligned with Mindset Theory. This cluster represents participants whose mindset talk is consistent and corresponds with their associated behaviors. We only had one participant in this cluster, P8, a female, first-year student who is not a CS major. P8 talked with a very strong and consistent growth mindset; 100% of her mindset talk was labeled with growth codes. For example, she expressed her belief that effort leads to improvement, saying *"I think I can become better, but I don't think that I am there just yet…I put in more hours, so it made me smarter at programming"*. Additionally, all but one of her associated behaviors were labeled with growth codes. For example, she expressed that she seeks out learning opportunities by frequently going to office hours: *"it's very interesting to see how [the TAs] think, because they're so much more experienced, to see how they look through a problem… my TAs are really good, they're really cool, they're pretty motivated, and it kinda motivates me to do well"*. Dweck notes that people with a fixed mindset tend to avoid more experienced people because they fear being compared to them and exposed as having lesser ability, while those with a growth mindset actively seek out opportunities to learn from people with more experience [12]. P8's growth mindset talk also aligns with her response to the survey and her self-identified mindset, which were both strongly growth mindset.

3.4.2 Misaligned Mindset and Associated Behaviors. This cluster describes participants whose associated behaviors are misaligned with their mindset talk. Participants in this cluster have a consistent mindset (75% of mindset talk was labeled with one mindset) but their associated behavior talk does not match (less than 75% of associated behaviors were labeled with the same mindset). The four participants in this cluster all presented growth mindsets through their talk, but over half of their associated behaviors were fixed.

As a representative example, consider P3, a female, first-year student who is a CS major. 100% of her mindset talk was labeled with growth codes. For example, she saw programming as something that requires learning rather than as an innate skill: *"[Programming is] more about learning the different structure, and the different strategies. I don't think everyone is just born with intuition for that. It's a lot of learning."* She also believes that if she works hard she can improve: *"If I studied really hard over the summer, I think I would be a lot better. Or at least I'd be more familiar with certain things than my peers."*

However, over half of her associated behaviors were labeled with fixed mindset codes. For example, when asked to share a time when she was proud of something she programmed, P3 decided to talk

about a time when she finished an assignment with ease: *"[I'm proud of the regular expressions programming assignment, that one wasn't too challenging. There wasn't much programming, but I got it done pretty quickly without that many errors, so I was pretty happy with myself."* Studies show that, when asked, people with fixed mindsets are more likely to talk about being proud of moments that demonstrate their ability, rather than their effort or learning [13]. In this case, P3 talked about being proud of a moment when she demonstrated her ability instead of one when she learned or overcame a challenge. Some of P3's associated behaviors were also labeled as growth mindset. For example, when describing a programming experience, she saw challenge as an opportunity to grow rather than as a negative reflection of ability when she said: *"I kept getting the error, and I would try and fix it, and then I would get more errors because of fixing that, well, trying to fix that error. And it was just really frustrating, but it was good practice"*. Hong et al. found that growth mindset students who were under-performing were more likely to seek out additional practice, tutorials, and remedial classes than students with a fixed mindset [24]. While P3's associated behaviors did not always align with her mindset, she filled out the mindset survey as growth and self-identified as growth. So the mindset survey and the self-report question captured her mindset but not her associated behaviors.

3.4.3 Mixed Mindset. The last cluster describes participants whose mindset talk is a mixture of the growth and fixed mindsets; between 25% and 75% of their mindset talk was coded as growth mindset. We categorized four participants as mixed mindset. Their percentage of associated behavior talk coded as growth varied widely, suggesting that the mixed mindset does not correlate with a specific associated behavior profile.

P2, a female, third-year student who is not a CS major, is an exemplary case study for this group. She frequently used both growth and fixed mindset talk, making seven and eight statements of each, respectively. She exhibited her mixed beliefs in her response to the question "Can anyone succeed in CS?", by saying *"I think there are people that are born for this and then there are people that need to try, but then if you try, if you really like it, I think you can. I think there is some advantage to those that their brains are wired in a way"*. Similar to her mixed beliefs, her associated behaviors were also mixed. She demonstrated motivation to continue working on the challenging programming task even after the clinical interview, saying *"No. I just want to figure this out"*. On the other hand, we noted five instances of avoidance behavior. For example, she said: *"before I get to other courses that are more fun, I have to go through the theory part of it, and since I'm not dedicated to CS, I don't want to put myself through the unnecessary hard work"*. Studies show that when asked to choose a type of problem to work on, fixed mindset people tend to choose problems that will demonstrate their ability, while growth mindset people pick challenging problems that will foster learning [13]. P2 responded to the survey slightly growth, which is surprising because her statements were relatively mixed or leaned fixed. But she did self-identify with both mindsets, saying: *"It's like, fixed, in a sense that I think there are people that are meant for it, and then not meant for it. But then, growth at the same time, because even if you're not meant for it, if you try hard enough and if you like it enough, you can always succeed in it. So fixed and growth."*

3.5 Self-Assessment Criteria Findings

Beyond coding the interview data to classify students' mindsets, we were also interested in identifying additional themes related to the ways that novice programmers talk about their intelligence. When analyzing the interviews, we noticed that students frequently assessed their own ability using a wide variety of criteria, which we call *self-assessment criteria*. For example, P9 mentioned the importance of memorizing syntax when he assessed his programming ability, saying *"I feel like I should remember the syntax for basic things, such as lists, and both C++ and Python, more closely than I currently do"*. P8 used the criteria that it is better to do work on your own when she said: *"I'm particularly proud of that [assignment] because I was able to figure out most of that on my own, and I didn't need as much TA help as I had anticipated"*. We identified seven different criteria that the nine participants used to evaluate programming ability. Table 2 describes the seven criteria codes.

3.6 Discussion

In our interview study, we found that only one of our participant's talk aligned with mindset research. The other eight participants fell into two categories: those whose behavior talk did not align with their mindset talk and those who expressed both growth and fixed mindsets. These findings may help explain some of the surprising and unexpected results of prior mindset research in computer science. Specifically, our results replicated Simon et al.'s findings that some students have mixed mindsets and provided new evidence that these students behave in a range of fixed and growth ways. Additionally, we show that the canonical mindset survey cannot capture mixed mindsets, as there is no response that indicates mixed beliefs. Our mixed mindset participants responded to the survey as growth mindset, fixed mindset, and in between. However, we found that when asked to self-identify with a mindset, all of the mixed mindset participants identified as both fixed and growth, suggesting that self-identification could be a more accurate measure of mindset than the survey. We also found that some students' behaviors were misaligned with their mindset talk, which may help to explain why Cutts et al.'s intervention successfully changed students' responses to mindset surveys but did not impact their associated behaviors. If students' behaviors are not always aligned with their mindset, we would not necessarily expect an intervention that successfully changes student mindsets to have an impact on their behaviors.

These findings are surprising because mindset theory is robust, and has been proven in many different domains and contexts. As a result, we suspect that other motivational factors may be interacting with mindset to produce these inconsistencies. We believe the frequent self-assessments and surprising self-assessment criteria we found could be one factor that interacts with mindsets. While researchers have mentioned the relationship between self-assessments and motivation in CS in previous work [26, 27], our study reveals specific self-assessment criteria that characterize the ways students evaluate programming-specific behaviors, like being able to memorize syntax or fix bugs quickly. These criteria emerged when students made assessments of their intelligence in the context of a programming experience. Such assessments, which are often called self-efficacy appraisals in the psychology literature, are particularly common when students are new to a field [3], like

	Code	Example Quote	Count
Identified in Interview Study	Better if you do it yourself	*"If I go to a... TA and I get a lot of help from them, then I feel kind of bad, because I didn't do the whole program by myself"*	1
	Better if you memorize syntax	*"They know various functions like the back of their hand"*	4
	Faster is better	*"If they can complete an assignment relatively quickly"*	19
	Code quality is important	*"Clean, understandable and short code"*	32
	Computer skills are important	*"By how fast they type"*	7
	Getting errors is bad	*"If it runs the first time they type it out"*	3
	Thinking and planning is not progress	*"If they keep typing and don't have to sit there and think"*	6
Identified in Survey Study	Correct solution	*"[Their] program works"*	5
	Decomposing problems is bad	*"If they can type out a whole long idea and tweak it as opposed to having to do each part piece by piece slowly"*	6
	Decomposing problems is good	*"If they do it in steps, checking/running their code as [they] go"*	7
	Thinking and planning is good	*"They are able to plan out and structure their thoughts on how to approach the code before writing it"*	13
	Good debugging skills	*"They are able to identify bugs... write test cases to check that their code is correct"*	12
	Good articulation skills	*"If they are able to stop and explain to you... what they are doing"*	11
	Ease of debugging	*"They understand how to debug a program quickly based off of first glance"*	15

Table 2: Codebook for the self-assessment criteria. The top set of codes were identified during the interview study. The bottom set of codes were identified in the survey study. Both sets of codes were used to code the open-ended survey question. The number in the right column represents the number of times each criterion was identified in the open-ended survey question.

our participants. Additionally, university students feel extra pressure because they have to choose a major, which may encourage more frequent self-efficacy appraisals as students consider their programming ability in their decision [15, 27].

We hypothesize that these self-efficacy appraisals may be one factor that interacts with student mindsets in CS. If students feel pressure to assess their own ability, they may choose to behave in ways that allow them to make self-assessments, rather than in ways that align with their mindsets about intelligence. Furthermore, these behaviors may depend on the criteria they believe are indicative of programming ability. For example, a student who thinks that people who are smart at programming can solve problems on their own may try to assess her own ability by not asking for help or using resources even if she has a growth mindset. These self-assessment behaviors could conflict with mindset associated behaviors, and produce effects that do not align with mindset theory.

In the interview study, we found that participants used a wide variety of criteria to assess their ability. However, we only interviewed nine students, and therefore do not know whether these findings generalize to a larger population, or whether students disagree about how to define and assess programming ability. While we think there might be a relationship between self-assessment criteria, mindset, and programming behaviors, we first need to understand the criteria in more depth before we can study the possible relationship. Therefore, we designed a second study to develop a deeper understanding of the self-assessment criteria, independent of mindset or programming behaviors.

4 SURVEY STUDY

In this second study, we further explore the self-assessment criteria to understand (1) whether these same criteria exist in a larger sample of students, (2) whether other self-assessment criteria arise that we did not find in the interview study and (3) whether there is consistency or variation in the criteria students use to measure intelligence in CS. Note that we do not aim to study the relationship between self-assessment criteria, mindsets, and programming behaviors in this study; we leave this for future work. To answer our questions, we designed a survey with three parts: an open-ended question about how students assess programming intelligence, 36 forced-choice Likert-scale questions about specific self-assessment criteria, and a mindset survey. We collected data from 103 novice CS undergraduates through two iterative rounds. In the first iteration, students answered the open-ended question and the mindset survey questions. In the second iteration, students also responded to Likert-scale questions about the specific self-assessment criteria that arose during the first iteration. This iterative design allowed us to explore the prevalence of a wide set of self-assessment criteria and better understand how students describe the criteria.

4.1 Participants and Setting

We recruited participants from the CS1 course at a large private university through the course discussion board and department email list. We conducted the study during the final week of the quarter. On the first iteration of the survey, we received 50 responses. On the second iteration of the survey, we received 56 responses, but discarded three who answered incorrectly to a check question, resulting in 53 usable responses. Of the participants we kept in our sample, 44% were female and 25% were CS majors. This closely represents the demographics of the class, which was 40% female and 18% CS majors. Participants who completed the survey were entered into a raffle for one of five $20 gift cards in each iteration.

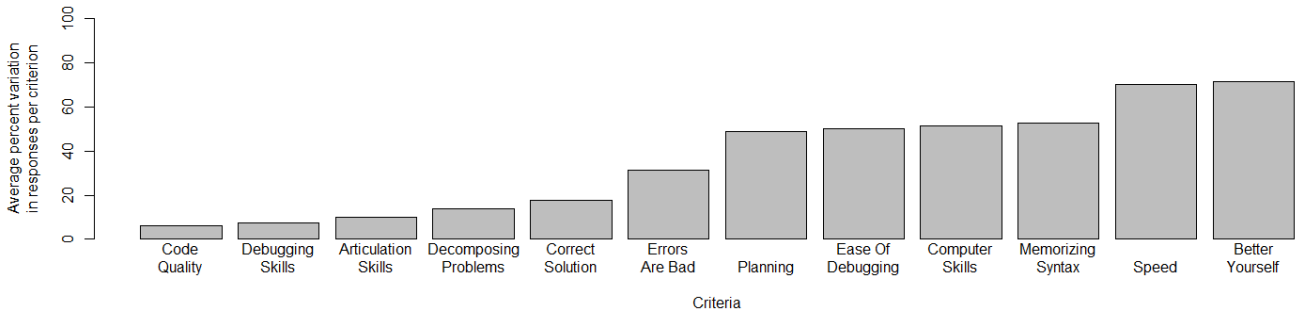

Figure 2: Variation in participant responses to Likert-scale survey questions, averaged across the three survey questions for each criterion. The variation is calculated by splitting the responses into two groups, agree and disagree, and then computing the inverse of the percent difference of the number of agrees and disagrees: ($1 - abs(agree - disagree)/(agree + disagree)$).

4.2 Open-ended Survey Question

We used an open-ended survey question to elicit self-assessment criteria from students, with the goals of confirming whether the criteria we observed in interviews are common and identifying new criteria. We asked students to respond to the following question: *"When watching someone program, how do you know if they are good at programming?"* To design this question, we informally tested a few options, including ones that directly asked how students evaluate themselves, but found that participants elaborated on assessment criteria most when asked about a specific instance of another person programming. Since students often compare themselves to peers when making self-assessments, we believe this question effectively elicits the criteria that our participants think are important for determining programming ability. The open-ended structure allows for free response and elaboration, without biasing responses by suggesting particular criteria.

4.2.1 Analysis. We qualitatively coded the responses to the open-ended question for all 103 participants using a combination of inductive and deductive methods [31, 39, 41]. First, we deductively coded the responses using the self-assessment criteria codes identified in the interview study. Then, we inductively coded the responses to identify new emerging themes. The two authors iteratively discussed and refined the codebook, and then each independently coded 10 survey responses (20% of the data). To check inter-rater reliability, we calculated a pooled, prevalence adjusted kappa of 97.5%, signifying excellent agreement [7, 10, 22]. The first author then coded the remaining 80% of the data.

4.2.2 Findings. Our analysis of the open-ended survey question revealed instances of participants using all seven of the self-assessment criteria identified in the interview study to evaluate programming ability. We also found seven new emergent criteria. The full codebook of criteria can be found in Table 2, along with example quotes from students' responses.

Interestingly, two of the new criteria are opposites of ones identified in the interview study, suggesting that participants disagree about these criteria. For example, some participants expressed that decomposing problems is good (*"if they do it in steps, checking/running their code as [they] go"*), while others expressed that decomposing problems is bad (*"they... think about it quickly and write it all in one sequence after thinking"*). However, the converses of the other criteria rarely or never came up. For example, only one participant indicated that using resources is an important part of coding, which could be considered a converse of the code *better if you do it yourself*. These findings suggest that our participants may agree about some self-assessment criteria, but disagree about others. However, given the nature of open-ended questions, we can not know if participants agree or disagree with a criterion unless they explicitly mention it, since the absence of a criterion does not necessarily imply disagreement. To test for disagreement in the criteria, we conducted a second iteration of the survey, in which we added Likert-scale questions that directly ask participants about their perspectives on the criteria.

4.3 Likert Scale Survey Questions

We designed a set of forced-choice Likert-scale questions to measure whether our participants agreed with statements related to the 14 self-assessment criteria that we identified through the interview study and first iteration of the survey study (see Table 2). We designed three questions for each criterion; two that expressed the criterion and one that expressed the converse of the criterion, by building on the quotes and code definitions from our previous studies. The two pairs of criteria that were opposites of each other were expressed through three questions rather than six, resulting in a total of 36 questions. We also included one check item that instructed students to answer 'disagree' to confirm that they carefully read the questions. We conducted think-alouds with students to test if the questions were clear and elicited the desired constructs [16, 33]. An example question for the criterion *faster is better* is: *"If you are faster at solving programming problems, then you are more intelligent at programming"*. An example of a question for *ease of debugging* is: *"Being able to fix a bug easily is an indication of programming intelligence"*. The Likert-scale questions were given to the 53 participants in our second round of testing.

4.3.1 Analysis. To analyze the Likert-scale data, we first flipped the responses to the converse questions, so that all numerical responses represented agreement with the criteria. Then, we looked at the distribution of responses to each question using bar graphs similar to the one shown in Figure 3. To understand the variation in participant responses, we split the responses into two groups,

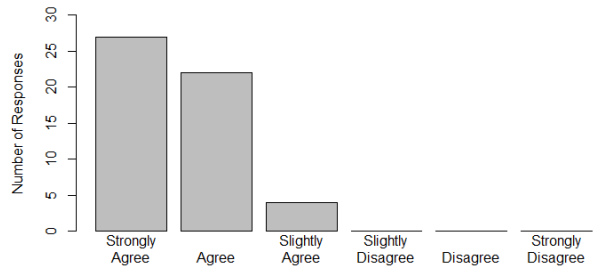

Figure 3: Histogram of responses to the Likert-scale survey question: *Being able to explain your program is an indication of programming intelligence.* All participants agreed with this statement.

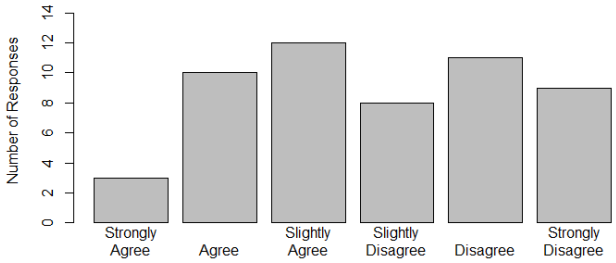

Figure 4: Histogram of responses to the Likert-scale survey question: *Someone is more intelligent at programming if they do an assignment on their own, rather than getting help to solve it.* Participants' responses are bimodally distributed, ranging from strongly disagree to strongly agree.

agree and disagree. Then, we calculated the inverse of the percent difference in number of agrees and disagrees to capture the amount of variation in responses. For example, given 20 survey responses in which 10 participants disagree with an item and 10 agree with an item, we would calculate $1 - abs(10 - 10)/20 = 1$, representing the maximum possible variation. However, given 20 participants who agree with an item and 0 who disagree, we would calculate $1 - abs(20 - 0)/20 = 0$, representing the minimum possible variation.

4.3.2 Findings. We found that participants consistently agreed with some self-assessment criteria, but expressed major disagreement in response to other criteria. Figure 2, shows the average variation in student responses to the three forced-choice Likert scale questions for each criterion. Most of our participants agreed with three of the self-assessment criteria; *good articulation skills*, *good debugging skills*, and *code quality is important*, which each had less than 10 percent variation in responses on average. This shows that our participants consistently thought that these skills demonstrate programming ability, perhaps because they are all discussed and encouraged in early stages of the CS curriculum. For example, Figure 3 shows the histogram of responses to one of the *good articulation skills* survey questions, demonstrating that all of the students agreed with the statement. On the other hand, six of the self-assessment criteria had over 50 percent variation in responses, indicating that there are differences in the way students define and measure programming intelligence. For example, Figure 4 shows the histogram of responses to one of the *better if you do it on your own* survey questions, demonstrating that participants had a wide range of beliefs.

While this study was not designed to uncover the relationship between self-assessment criteria and mindsets, we were interested in measuring whether any criteria were correlated with a particular mindset. However less than 5% of participants reported fixed mindsets on the survey, so we did not have enough data to explore this question. Given the variation in student responses, it is clear that growth mindset students do not all agree with the same self-assessment criteria, so the criteria students use could be one factor that interacts with mindsets to influence programming behaviors.

5 CONCLUSIONS

We present the results of two studies: an interview study where students worked on a challenging programming task and then discussed their beliefs about programming intelligence, and a survey study that asked about the criteria that students use to evaluate programming intelligence. In the interviews, we found that only one participant's talk aligned with mindset theory; the other eight participants' talk either included both fixed and growth attributes or misaligned with their associated behaviors. This is surprising given the robustness of mindset research, suggesting there is something else influencing the enactment of mindsets and associated behaviors in the domain of CS. During the interviews, we also found that students frequently made self-efficacy appraisals using a variety of criteria. Our findings in the survey study confirm that students define and measure programming intelligence in different ways. We suggest that these criteria may interact with students' mindsets and influence their behaviors. These self-assessment criteria could have a particularly strong impact on university CS students because they frequently make self-efficacy appraisals while deciding whether to pursue a major or career in CS.

While these initial results provide valuable insights about mindsets in CS, there are a number of limitations that we hope to address in future work. First, our interview study had a small number of participants, so we do not know how the mindset clusters that we identified will generalize. For both studies, we recruited participants from the same institution, so we do not know if there are environmental factors influencing the results. Finally, our analysis depends on qualitatively coding students' talk rather than directly analyzing their behavior during programming, and it is possible that students' talk does not always reflect their behavior in practice.

In future work, we plan to study the relationship between students' self-assessment criteria, mindsets, and programming behaviors to improve our understanding of how motivational factors impact persistence. This work takes an important first step in this direction by establishing the existence of self-assessment criteria and demonstrating the nuances of mindsets in CS. We hope this will spark additional research with the ultimate goal of designing interventions that motivate students to persist in CS.

REFERENCES

[1] 2018. Occupational Outlook Handbook. (April 2018). https://www.bls.gov/ooh/fastest-growing.htm

[2] Joshua Aronson, Carrie B. Fried, and Catherine Good. 2002. Reducing the Effects of Stereotype Threat on African American College Students by Shaping Theories of Intelligence. *Journal of Experimental Social Psychology* 38, 2 (March 2002), 113–125. https://doi.org/10.1006/jesp.2001.1491

[3] Albert Bandura. 1982. Self-efficacy mechanism in human agency. *American psychologist* 37, 2 (1982), 122.

[4] Theresa Beaubouef and John Mason. 2005. Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin* 37, 2 (June 2005), 103. https://doi.org/10.1145/1083431.1083474

[5] Jens Bennedsen and Michael E. Caspersen. 2007. Failure rates in introductory programming. *ACM SIGcSE Bulletin* 39, 2 (2007), 32–36.

[6] Lisa S. Blackwell, Kali H. Trzesniewski, and Carol S. Dweck. 2007. Implicit theories of intelligence predict achievement across an adolescent transition: A longitudinal study and an intervention. *Child development* 78, 1 (2007), 246–263. http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8624.2007.00995.x/full

[7] Ted Byrt, Janet Bishop, and John B. Carlin. 1993. Bias, prevalence and kappa. *Journal of Clinical Epidemiology* 46, 5 (May 1993), 423–429. https://doi.org/10.1016/0895-4356(93)90018-V

[8] J. M. Cohoon. 2006. Just get over it or just get on with it. *Retaining women in undergraduate computing. In J. Cohoon & W. Aspray (Eds.), Women and information technology: Research on underrepresentation* (2006), 205–238.

[9] Quintin Cutts, Emily Cutts, Stephen Draper, Patrick O'Donnell, and Peter Saffrey. 2010. Manipulating mindset to positively influence introductory programming performance. In *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, 431–435. http://dl.acm.org/citation.cfm?id=1734409

[10] Han De Vries, Marc N. Elliott, David E. Kanouse, and Stephanie S. Teleki. 2008. Using Pooled Kappa to Summarize Interrater Agreement across Many Items. *Field Methods* 20, 3 (Aug. 2008), 272–282. https://doi.org/10.1177/1525822X08317166

[11] Carol S. Dweck. 1999. *Self-theories: Their role in motivation, personality, and development*. Psychology Press.

[12] Carol S Dweck. 2006. *Mindset: The new psychology of success*. Random House Incorporated.

[13] Carol S Dweck and Janine Bempechat. 1983. Children's theories of intelligence: Consequences for learning. *Learning and motivation in the classroom* (1983), 239–256.

[14] Carol S. Dweck and Ellen L. Leggett. 1988. A social-cognitive approach to motivation and personality. *Psychological review* 95, 2 (1988), 256.

[15] Jacquelynne S. Eccles and Bonnie L. Barber. 1999. Student council, volunteering, basketball, or marching band what kind of extracurricular involvement matters? *Journal of adolescent research* 14, 1 (1999), 10–43. http://jar.sagepub.com/content/14/1/10.short

[16] K Anders Ericsson and Herbert A Simon. 1984. *Protocol analysis: Verbal reports as Data*. MIT Press, Cambridge, MA.

[17] Allan Fisher and Jane Margolis. 2002. Unlocking the clubhouse: the Carnegie Mellon experience. *ACM SIGCSE Bulletin* 34, 2 (2002), 79–83. http://dl.acm.org/citation.cfm?id=543836

[18] Abraham E. Flanigan, Markeya S. Peteranetz, Duane F. Shell, and Leen-Kiat Soh. 2015. Exploring Changes in Computer Science Students' Implicit Theories of Intelligence Across the Semester. ACM Press, 161–168. https://doi.org/10.1145/2787622.2787722

[19] Herbert Ginsburg. 1997. *Entering the child's mind: The clinical interview in psychological research and practice*. Cambridge University Press, New York.

[20] Catherine Good, Joshua Aronson, and Michael Inzlicht. 2003. Improving adolescents' standardized test performance: An intervention to reduce the effects of stereotype threat. *Journal of Applied Developmental Psychology* 24, 6 (Dec. 2003), 645–662. https://doi.org/10.1016/j.appdev.2003.09.002

[21] Elizabeth A. Gunderson, Sarah J. Gripshover, Carissa Romero, Carol S. Dweck, Susan Goldin-Meadow, and Susan C. Levine. 2013. Parent Praise to 1- to 3-Year-Olds Predicts Children's Motivational Frameworks 5 Years Later. *Child Development* 84, 5 (Sept. 2013), 1526–1541. https://doi.org/10.1111/cdev.12064

[22] Kevin A. Hallgren. 2012. Computing inter-rater reliability for observational data: an overview and tutorial. *Tutorials in quantitative methods for psychology* 8, 1 (2012), 23.

[23] Gail D. Heyman and Carol S. Dweck. 1998. Children's thinking about traits: Implications for judgments of the self and others. *Child development* 69, 2 (1998), 391–403.

[24] Ying-yi Hong, Chi-yue Chiu, Carol S. Dweck, Derrick M.-S. Lin, and Wendy Wan. 1999. Implicit theories, attributions, and coping: A meaning system approach. *Journal of Personality and Social psychology* 77, 3 (1999), 588. http://psycnet.apa.org/journals/psp/77/3/588/

[25] Antti-Juhani Kaijanaho and Ville Tirronen. 2018. Fixed versus Growth Mindset Does not Seem to Matter Much: A Prospective Observational Study in Two Late Bachelor level Computer Science Courses. In *Proceedings of the 2018 ACM Conference on International Computing Education Research - ICER '18*. ACM Press, Espoo, Finland, 11–20. https://doi.org/10.1145/3230977.3230982

[26] Päivi Kinnunen and Beth Simon. 2012. My program is ok —am I? Computing freshmen's experiences of doing programming assignments. *Computer Science Education* 22, 1 (March 2012), 1–28. https://doi.org/10.1080/08993408.2012.655091

[27] Colleen M. Lewis, Ken Yasuhara, and Ruth E. Anderson. 2011. Deciding to major in computer science: a grounded theory of students' self-assessment of ability. In *Proceedings of the seventh international workshop on Computing education research*. ACM, 3–10.

[28] Alex Lishinski, Aman Yadav, Jon Good, and Richard Enbody. 2016. Learning to Program: Gender Differences and Interactive Effects of Students' Motivation, Goals, and Self-Efficacy on Performance. ACM Press, 211–220. https://doi.org/10.1145/2960310.2960329

[29] Dastyni Loksa and Andrew J. Ko. 2016. The Role of Self-Regulation in Programming Problem Solving Process and Success. ACM Press, 83–91. https://doi.org/10.1145/2960310.2960334

[30] Dastyni Loksa, Andrew J. Ko, Will Jernigan, Alannah Oleson, Christopher J. Mendez, and Margaret M. Burnett. 2016. Programming, Problem Solving, and Self-Awareness: Effects of Explicit Guidance. ACM Press, 1449–1461. https://doi.org/10.1145/2858036.2858252

[31] Matthew B Miles, A Michael Huberman, and Johnny Saldana. 2018. *Qualitative data analysis: A methods sourcebook*. Sage publications.

[32] Claudia M. Mueller and Carol S. Dweck. 1998. Praise for intelligence can undermine children's motivation and performance. *Journal of personality and social psychology* 75, 1 (1998), 33. http://psycnet.apa.org/journals/psp/75/1/33/

[33] Judith S Olson and Wendy A Kellogg. 2014. *Ways of Knowing in HCI*. Vol. 2. Springer.

[34] Eleanor O'Rourke, Kyla Haimovitz, Christy Ballweber, Carol Dweck, and Zoran Popović. 2014. Brain points: a growth mindset incentive structure boosts persistence in an educational game. ACM Press, 3339–3348. https://doi.org/10.1145/2556288.2557157

[35] Vennila Ramalingam, Deborah LaBelle, and Susan Wiedenbeck. 2004. Self-efficacy and mental models in learning to program. In *ACM SIGCSE Bulletin*, Vol. 36. ACM, 171–175.

[36] Dale H. Schunk. 1989. Attributions and Perceptions of Efficacy during Self-Regulated Learning by Remedial Readers. (1989).

[37] Michael J. Scott and Gheorghita Ghinea. 2014. On the domain-specificity of mindsets: The relationship between aptitude beliefs and programming practice. *IEEE Transactions on Education* 57, 3 (2014), 169–174. http://ieeexplore.ieee.org/abstract/document/6662493/

[38] Beth Simon, Brian Hanks, Laurie Murphy, Sue Fitzgerald, Renée McCauley, Lynda Thomas, and Carol Zander. 2008. Saying isn't necessarily believing: influencing self-theories in computing. In *Proceedings of the Fourth international Workshop on Computing Education Research*. ACM, 173–184. http://dl.acm.org/citation.cfm?id=1404537

[39] Anselm Strauss and Juliet Corbin. 1994. Grounded theory methodology. *Handbook of qualitative research* 17 (1994), 273–85.

[40] F. Boray Tek, Kristin S. Benli, and Ezgi Deveci. 2018. Implicit Theories and Self-Efficacy in an Introductory Programming Course. *IEEE Transactions on Education* 61, 3 (Aug. 2018), 218–225. https://doi.org/10.1109/TE.2017.2789183

[41] David R Thomas. 2006. A general inductive approach for analyzing qualitative evaluation data. *American journal of evaluation* 27, 2 (2006), 237–246.

[42] Nanette Veilleux, Rebecca Bates, Cheryl Allendoerfer, Diane Jones, Joyous Crawford, and Tamara Floyd Smith. 2013. The relationship between belonging and ability in computer science. In *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, 65–70.

[43] Rebecca Vivian, Katrina Falkner, and Nickolas Falkner. 2013. Computer science students' causal attributions for successful and unsuccessful outcomes in programming assignments. In *Proceedings of the 13th Koli Calling International Conference on Computing Education Research - Koli Calling '13*. ACM Press, Koli, Finland, 125–134. https://doi.org/10.1145/2526968.2526982

[44] David S. Yeager, Carissa Romero, Dave Paunesku, Christopher S. Hulleman, Barbara Schneider, Cintia Hinojosa, Hae Yeon Lee, Joseph O'Brien, Kate Flint, Alice Roberts, Jill Trott, Daniel Greene, Gregory M. Walton, and Carol S. Dweck. 2016. Using design thinking to improve psychological interventions: The case of the growth mindset during the transition to high school. *Journal of Educational Psychology* 108, 3 (2016), 374–391. https://doi.org/10.1037/edu0000098