ABSTRACT

VANDENBERG, JESSICA LEIGH. Design, Validation, and Application of An Upper Elementary Computer Science Attitudes Survey: A Mixed Methods Approach. (Under the direction of Dr. John Nietfeld and Dr. Eric Wiebe).

Exposing students to computing activities, especially in younger grades, increases the likelihood that students will develop an interest in computer science (CS). This interest may translate into a heightened ability that carries over into elective middle and high school courses, selecting a college major, and eventually a career. Upper elementary students (typically ages 9 to 11) are at a critical time in developing their interests and in receiving messages that inform how they feel about their abilities. Expectancy-value theory makes use of Bandura's *self-efficacy* theory—the belief a student has in his or her ability to complete a task successfully—and the notion of *outcome expectancy*—students' belief that their behaviors will result in a desired outcome. Together, these help students form proximal and distal goals. These distal goals, be they future academic choices or career options, are influenced by students' expectations of success and how much they value, or show interest in, the task. Assessing young students' CS interests, through an attitudinal survey focused on self-efficacy and outcome expectancy items, is relatively novel.

Students' perceptions of themselves—including whether they belong in a class, a major, or a field—and their abilities likely affects to what extent they are successful in regulating their learning in that domain. High self-efficacy students persist when challenges arise and they utilize more effective strategies while learning; moreover, they tend to modify their goals and strategy use through feedback and engage in adaptive help seeking. Because learning is a social process, it is also important to explore how student self-efficacy and outcome expectancy interacts with the dynamics of academic collaboration. This is a particularly interesting area to study in CS

education due to the growing popularity of collaborative (pair) programming. As an analytic approach, students' discourse as they problem solve is a rich source of data on collaborative regulation because it is from what students say that we can gain insight into what they think, want, and make sense of the task.

This dissertation research sought to address the lack of validated instruments for assessing young students' computing interests, through both qualitative and quantitative methods. Further, the validated instrument was then used within a classroom-based study, underscoring its pragmatic and empirical uses. The organization of this dissertation follows a three-article style approach, with each article theoretically and empirically linked to the next.

Findings from the qualitative validation of the instrument indicated that upper elementary students were unable to respond appropriately to our initial set of items, resulting in several significant modifications through three iterative studies. Having determined the items were qualitatively valid, the instrument underwent classical test theory and item response theory-Rasch validation and reliability analysis. The instrument was determined to be largely psychometrically bias-free, and, in alignment with literature, males had higher CS self-efficacy and outcome expectancy beliefs than females. Validity established; the instrument was used as one of two major measures in an exploration of classroom-based dyadic discourse. Of the dyads examined in the final study, two demonstrated anticipated regulatory behaviors and collaborative discourse, with the remaining offering more diversity in how students collaborate. Cross-case analyses revealed a range of ways the dyads' self-efficacy and CS conceptual understanding affected their collaborative discourse. Recommendations for practitioners and researchers are provided.

Design, Validation, and Application of An Upper Elementary Computer Science Attitudes Survey: A Mixed Methods Approach

by
Jessica Leigh Vandenberg

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Teacher Education and Learning Sciences

Raleigh, North Carolina
2021

APPROVED BY:

Dr. John Nietfeld Committee Co-Chair	Dr. Eric Wiebe Committee Co-Chair
Dr. Soonhye Park	Dr. Steven Porter

BIOGRAPHY

Jessica Vandenberg earned her Master's in Teaching in 2005 and then spent 11 years as a classroom teacher, teaching all grades K-6, all subjects, and working with gifted and ELL students. She first encountered computer science and coding after facilitating a unit of study on philosophy and symbolic logic during the 2005-2006 school year. It would be many years before she returned to this domain.

Jessica enrolled in the Educational Psychology PhD program full-time in fall of 2016 and joined the research lab under Dr. Eric Wiebe housed at the Friday Institute for Educational Innovation in spring of 2018. It is here that she was able to combine her personal and professional experiences as a teacher with her nascent interests in computing.

ACKNOWLEDGMENTS

I thank my committee for their patience, invaluable feedback, and personal and professional support. I have learned from each of you; from your classroom instruction to office hours to lengthy email threads to countless Zoom meetings, I thank you for your time and wisdom.

I thank my research lab mates for their willingness to be sounding boards and helping me move all those laptops around. My co-authors, Jen Tsan, Zarifa Zakaria, Danielle Boulden, and Arif Rachmatullah, are amazing humans and writers without whom these chapters would not have occurred. Jen, you are forever my academic other half and car ride/study implementations will never be the same.

Thank you to friends and fellow PhD graduates, Cody Smith, Whitney McCoy, Sarah Karamarkovich, and Sara Egan Warren, for striking that perfect balance of checking in on my progress and providing nudges when needed.

And thank you Ross for listening, asking, supporting, and for the words and stickers of encouragement I have found among my notes.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Introduction

Exposing students to computing activities, especially in younger grades, increases the likelihood that students will develop an interest in computer science (CS). This interest may translate into a heightened ability that carries over into elective middle and high school courses, selecting a college major, and eventually a career. Upper elementary students (typically ages 9 to 11) are at a critical time in developing their interests and in receiving messages that inform how they feel about their abilities. Expectancy-value theory (Eccles & Wigfield, 2002; Wigfield & Eccles, 2000) makes use of Bandura's (1986) *self-efficacy* theory—the belief a student has in his or her ability to complete a task successfully—and the notion of *outcome expectancy*—students' belief that their behaviors will result in a desired outcome. Together, these help students form proximal and distal goals. These distal goals, be they future academic choices or career options, are influenced by students' expectations of success and how much they value, or show interest in, the task (Eccles & Wigfield, 2002).

Assessing young students' CS interests, through an attitudinal survey focused on self-efficacy and outcome expectancy items, is relatively novel. Kukul, Gökçearslan, and Günbatar (2017) assessed middle school students' (aged 12 through 14) self-efficacy for very specific computing concepts. Kong et al. (2018) and Mason and Rich (2020) assessed elementary school students' (grades 4 through 6) self-efficacy in programming/coding. There is a definite need for an assessment that covers both self-efficacy and outcome expectancies for upper elementary students, and which has been psychometrically tested for gender and/or racial bias. This latter notion, regarding the potential for bias, is extremely important to explore, especially in a domain with a history of exclusion. CS is one such domain. It is important not only for CS learning environments to be inclusive but also for instruments used for assessing student performance and perceptions to be bias free, especially if the results of those instruments are used to inform policy and practice.

Students' perceptions of themselves—including whether they belong in a class, a major, or a field—and their abilities likely affects to what extent they are successful in regulating their learning in that domain (Pajares, 2002). High self-efficacy students persist when challenges arise and they utilize more effective strategies while learning (Artino, 2012); moreover, they tend to modify their goals and strategy use through feedback (Butler & Winne, 1995) and engage in adaptive help seeking (Ryan & Shin, 2011). Because learning is a social process, it is also important to explore how student self-efficacy and outcome expectancy interacts with the dynamics of academic collaboration. This is a particularly interesting area to study in CS education due to the growing popularity of collaborative (pair) programming. As an analytic approach, students' discourse as they problem solve is a rich source of data on collaborative regulation because it is from what students say that we can gain insight into what they think, want, and make sense of the task (Johnstone, 2017; Potter, 1998).

Statement of the Problem

Students' affective states, including interest, play a role in their cognitive and learning processes (Baker et al., 2010; Reeve et al., 2015). If interest drives learning, then lack of interest likely forestalls learning. This, according to Schmidt (2011) creates a cycle in which disinterested students do not build their capacity in a certain domain, which leads to a decrease in self-efficacy and a further reduction in interest. This may be all the more critical when students are younger and experiences in a range of domains support or hamper their sense of self-efficacy. Girls consistently underestimate their abilities in STEM subjects (Eccles, 1987). Females and historically underrepresented minorities (URM) often express less confidence in their CS abilities compared to white male counterparts (Litzler et al., 2014). Similarly, little is known about how these affective states influence collaborative behaviors, especially for younger students.

Purpose of the Research

The purpose of this inquiry is to explore the range and uses of computer science attitudes upper elementary students hold. An emphasis will be placed on students' own words and their self-reports of their attitudes, self-efficacy, and outcome expectancy in CS. This mixed methods study aims to further the understanding of what CS words students know and how they conceptualize CS concepts, validate a refined measure, and model students' discourse around CS self-efficacy and outcome expectancy.

Research Objectives

To achieve the purpose written above, the following three overarching research objectives have been developed. The findings and conclusions of this research will be presented in a series of article manuscripts. The research objectives guiding the three articles are:

Research Objective #1: The refinement of a validated survey to measure upper elementary students' attitudes and perspectives about computer science.

Research Objective #2: The validation of the Elementary CS Attitudes instrument refined through RO #1.

Research Objective #3: The exploration of qualitative discourse data and quantitative CS attitudes and conceptual understanding survey data using dynamic network models.

Significance of the Research

Young students' attitudes toward CS likely reflect their interests, and more specifically, their self-efficacy, and that their behaviors will result in a certain outcome (outcome expectancy). There are no studies, to date, that make use of cognitive interviews to ascertain students' feedback on appropriate wording of items, that then validates the instrument using the newly worded items, and then uses the measure in a mixed methods study leveraging student discourse. This study privileged students—their input, word choice, self-reports, and discourse—throughout, holding that they are capable of reporting their interests and knowledge and that they are a reliable source of such information.

By knowing how young students perceive CS, researchers can better promote curricula or interventions aimed at bolstering students' CS interest and self-efficacy, all in hopes of building a diverse and sustainable STEM pipeline. Similarly, a better understanding of the interaction of affective states and collaborative work will help guide strategies for productive collaborative CS work amongst upper elementary students.

Definitions

To understand and clarify the terms used in this study, the following are defined here:

Self-efficacy

Bandura (1994) maintains that self-efficacy is an individual's belief in their ability to perform a task in pursuit of a goal. There are four sources of self-efficacy: mastery experiences, vicarious learning, social persuasion, and affective or physiological arousal. Students' sense of self-efficacy in a domain or task can change over time based on how they perceive they are succeeding, how they take in information from others who are modeling, how they respond to the words of others, and how they react to the clues their own bodies give. Individuals develop self-efficacy beginning as infants as they express agency over themselves and their environment. Families, peers, and school environments further provide challenge and context to the development of self-efficacy. Efficacy beliefs shape, to some extent, outcome expectancies (Pajares, 1997).

Outcome Expectancy

Generally, students who hold a higher belief in their ability to complete a task, expect success in that task; relatedly, those who lack confidence in their ability to complete a task will likely anticipate a poor outcome. Eccles (1983) and Wigfield and Eccles (1992) maintain that self-efficacy judgments interact with outcome expectancies by influencing which tasks or activities an individual will freely engage in. In other words, individuals tend to self-assess how they opt to spend their time by determining to what extent they feel capable of being successful.

References

Artino, A. R. (2012). Academic self-efficacy: from educational theory to instructional practice. *Perspectives on Medical Education, 1*(2), 76-85.

Baker, R. S., D'Mello, S. K., Rodrigo, M. M. T., & Graesser, A. C. (2010). Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies, 68*(4), 223-241.

Bandura, A. (1984). Recycling misconceptions of perceived self-efficacy. *Cognitive Therapy and Research, 8*, 231-255.

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Englewood Cliffs, NJ: Prentice-Hall.

Butler, D. L., & Winne, P. H. (1995). Feedback and self-regulated learning: A theoretical synthesis. *Review of Educational Research*, *65*(3), 245-281.

Eccles, J. (1983). Expectancies, values, and academic behaviors. In J. T. Spence (Ed.), *Achievement and achievement motives: Psychological and sociological approaches* (pp.75-146). San Francisco: Freeman.

Eccles, J. S. (1987). Gender roles and women's achievement-related decisions. *Psychology of Women Quarterly, 11*, 135–172.

Eccles, J. S., & Wigfield, A. (2002). Motivational beliefs, values, and goals. *Annual Review of Psychology*, *53*, 109-132.

Johnstone, B. (2017). *Discourse analysis*. John Wiley & Sons.

Kong, S. C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education*, *127*, 178-189.

Kukul, V., Gökçearslan, Ş., & Günbatar, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Eğitim Teknolojisi Kuram ve Uygulama [Educational Technology-Theory and Practice]*, *7*(1), 158-179.

Litzler, E., Samuelson, C. C., & Lorah, J. A. (2014). Breaking it down: Engineering student STEM confidence at the intersection of race/ethnicity and gender. *Research in Higher Education*, *55*(8), 810-832.

Mason, S. L., & Rich, P. J. (2020). Development and analysis of the Elementary Student Coding Attitudes Survey. *Computers & Education*, 103898.

Pajares, F. (1997). Current directions in self-efficacy research. Advances in motivation and achievement, 10(149), 1-49.

Pajares, F. (2002). Gender and perceived self-efficacy in self-regulated learning. *Theory Into Practice, 41*(2), 116-125.

Potter, J. (1998). Discursive social psychology: From attitudes to evaluative practices. *European Review of Social Psychology, 9*(1), 233-266.

Reeve, J., Lee, W., & Won, S. (2015). Interest as emotion, as affect, as schema. *Interest in mathematics and science learning*, 79-92.

Ryan, A. M., & Shin, H. (2011). Help-seeking tendencies during early adolescence: An examination of motivational correlates and consequences for achievement. *Learning and Instruction, 21*(2), 247-256.

Schmidt, F. (2011). A theory of sex differences in technical aptitude and some supporting evidence. *Perspectives on Psychological Science, 6*, 560–573.

Wigfield, A., & Eccles, J. (1992). The development of achievement task values: A theoretical analysis. *Developmental review*, *12*, 265-310.

Wigfield, A., & Eccles, J. S. (2000). Expectancy–value theory of achievement motivation. *Contemporary Educational Psychology, 25*(1), 68-81.

Wilson, K., & Narayan, A. (2016). Relationships among individual task self-efficacy, self-regulated learning strategy use and academic performance in a computer-supported collaborative learning environment. Educational Psychology, 36(2), 236-253.

CHAPTER 2: ELEMENTARY STUDENTS' UNDERSTANDING OF CS TERMS

A version of this chapter appears as:

Vandenberg, J., Tsan, J., Boulden, D., Zakaria, Z., Lynch, C., Boyer, K. E., & Wiebe, E. (2020). Elementary students' understanding of CS terms. *ACM Transactions on Computing Education (TOCE)*, *20*(3), 1-19.

Abstract

The language and concepts used by curriculum designers is not always interpreted by children as designers intended. This can be problematic when researchers use self-reported survey instruments in concert with curricula, which often rely on the implicit belief that students' understanding aligns with their own. We report on our refinement of a validated survey to measure upper elementary students' attitudes and perspectives about computer science (CS), using an iterative, design-based research approach informed by educational and psychological cognitive interview processes. We interviewed six groups of students over three iterations of the instrument on their understanding of CS concepts and attitudes toward coding. Our findings indicated that students could not explain the terms *computer programs* nor *computer science* as expected. Furthermore, they struggled to understand how coding may support their learning in other domains. These results may guide the development of appropriate CS-related survey instruments and curricular materials for K-6 students.

Introduction

Researchers have found that negative stereotypes about computer science influence students' decisions to pursue or abandon a degree in the field (Baker et al., 2010; Lewis et al., 2016). Holding a positive attitude towards computer programming is likewise correlated with higher self-efficacy in programming (Özyurt & Özyurt, 2015). Students acquire their beliefs about topics from their first-hand experiences, from direct observation, and from evaluating what others have told them (Azjen & Cote, 2008). These beliefs form the basis of their attitudes, which in turn impart meaning to objects or activities (Schwarz, 2007).

With initiatives such as Hour of Code[1] and communities like CS For All[2], students are now being introduced to computer science concepts as early as elementary school. Indeed, the new K-12 Computer Science Framework specifically emphasizes the need for young students to engage in varied types of computing (Alano et al., 2018). In light of this instructional push, researchers need to focus on understanding elementary students' beliefs, attitudes, and experiences in computer science. Nine to eleven-year-old students are within Piaget's (2002) concrete operational stage; the majority of them should have the ability to think logically and to recognize and be able to share their unique opinions of their beliefs regarding interventions such as these.

Many of the existing curricular interventions and associated survey items for elementary-age students have been developed by domain experts who often contextualize their conceptual and technical terminology at the adult level, which they in turn expect students to master. However, children are not always comfortable with these terms nor do they understand general concepts like 'programming' in the way that the curriculum designers intend. Many researchers use self-report and attitudinal survey instruments with the implicit belief that the students' understanding of the terms and concepts matches their own. This mismatch may lead researchers to come to incorrect conclusions as to what students' experiences and attitudes are with regards to computer science and to the efficacy of their interventions. A lack of familiarity with key terms that anchor self-report items can lead to instability and an absence of consensus among

[1] https://hourofcode.com/us
[2] https://www.csforall.org/

students regarding the meaning of a word or phrase. This disconnect can then result in unpredictable shifts in a student's understanding of a survey item (gamma shift) or how they scale their response to the item (beta shift) pre to post-intervention (Broderson & Thornton, 2011).

One way to access students' beliefs is to ask them (Schaeffer & Presser, 2003). The cognitive interviewing process, as detailed in Karabenick et al. (2007), probes students on their understanding of what the item means and which answer they would select, in addition to other related probes. Cognitive interviewing is an iterative process in which findings from one phase necessitate refinement and further testing, with instrument development being an outcome.

In this paper, we assess the students' attitudes toward and understanding of computer science (CS) concepts through a series of cognitive interviews. As part of this process we qualitatively evaluated their responses to specific items which then guided changes in the wording and number of items. Our analyses indicate that 4th and 5th grade students broadly understand computer science as coding. They are most comfortable using coding to describe what they do, for example, while making a game on Scratch[3] or giving a robot like Sphero™ the right directions. Some students in our survey were able to make clear delineations between writing/building code and debugging, but many students struggled to connect coding to other subjects (i.e. science, mathematics, engineering) studied in school. These findings can inform the development and design of survey instruments as well as decisions on appropriate vocabulary to use to support elementary students in learning computer science.

[3] https://scratch.mit.edu/

Related Work

Cognitive Interviewing with Youth

Our current work is influenced by prior research on interviewing survey-takers about their interpretation of survey items. Schwarz (1999) notes that often adults struggle to comprehend the meaning of an item and that context also influences self-reports of attitude. Cognitive interviews to validate survey items are less common with children, despite important developmental differences in comprehension, understanding of abstract concepts, and working memory (Woolley et al., 2004).

Cognitive interviewing as part of survey development increases the likelihood that self-report survey items are valid (Karabenick et al., 2007). Cognitive validity refers to how well a respondent's thought processes align with what the survey designer intended. In other words, the goal is to determine to what extent the student thinks about and responds to the item as the designer intended. This is a layered, multi-stage process; students must read and interpret the item, determine the intent and keep this information in working memory, connect experiences from memory to the item's intent, read and interpret the answer choices, combine their inferred meaning with their own personal experiences, and then finally select their answer choice (Karabenick et al., 2007). Cognitive interviewing allows us to probe students on their thinking during any part of that process.

Researchers interested in developing valid measures have utilized cognitive interviewing processes with children and adolescents in a range of topic areas. Woolley et al. (2004) interviewed third through fifth grade students to validate a drug abuse prevention measure. Over the course of two phases of interviews, they found that wording of several items required modification to reflect a more concrete interpretation and that answer choices likewise needed to be more objective. Arthur et al. (2002) took a similar approach with high school students. They were cognitively interviewed on substance abuse and risk-taking behaviors in order to develop an instrument assessing risk and protective factors affecting adolescents. The team had students think aloud during the entire interview process, with final results indicating that almost 100 of the 350 possible items were unclear.

In a large European health study, the authors used cognitive interviews with children and adolescents to develop condition-specific health-related measures. These measures were nested within modules that often included multiple questionnaires focused on different aspects of the disease, the emotional implications of the illness, and surveys were intended for both the patient and the caregiver (Baars et al., 2005). The results from this cross-national study indicated that the cognitive interview process helped inform the researchers about the relevance, coherency, and appropriateness of the content for each condition-specific health module. All of these studies highlight the need to carefully listen to survey respondents and modify wording, sometimes several times, or outright eliminate items to ensure that children and adolescents can read, understand, and answer survey items.

Affective Research in CS

Affective research in CS has sought to understand the diversity of issues impacting students' interest in CS and the field in general. This is of interest in part because students' affect influences their cognition and learning processes (Baker et al., 2010). Study foci include students' feelings of belongingness within the field and the need to counteract stereotypes (Lewis et al., 2016), students' inaccurate preconceptions (Hewner, 2013) or misconceptions (Grover et al., 2014) about CS, and students' range of positive and negative experiences with CS classes

(Hewner & Guzdial., 2008). A comprehensive literature review of affective computing indicated that students readily recognize their own and others' emotions (Reis et al., 2018).

One mechanism for assessing affect is self-report measures (Graesser et al., 2006). These self-reports give information about how the student perceives his or her emotions at a given time and in response to a task, event, or prompt. A dearth of existing validated affective instruments specific to CS has meant that some researchers have used self-efficacy instruments contextualized in other disciplines, such as mathematics, as a proxy for measuring the impact of CS-related interventions (e.g., Psycharis & Kallia, 2017). To address this lack of appropriate CS-specific instruments, Tsai, Wang and Hsu (2019) recently developed a self-report instrument for measuring self-efficacy for computer programming. Although they state the instrument can be used for students older than middle school, the validation was conducted with a sample of college students. A similar validation effort was done on a self-efficacy scale in Turkish with secondary school students age 12-14 (Kukul et al., 2017).

Moving beyond just utilizing Likert-type self-report, Weintrop's (2016) work with high school students in three different programming environments highlights the value of asking students their perceptions of and experiences with programming. In his work, students typed open-ended survey responses and spoken interview responses were analyzed in tandem as a way of detailing students' conceptions of programming and changes over the course of the study. The work reviewed briefly above underscores the interest in how students perceived CS and how those perceptions affected their work in CS. Our instrument development effort focuses on a younger group of students, and although the published studies inform our work, they only provide a starting point for how we should word future survey instruments for this population.

Sociocultural Theory

Children learn and develop when external—social and cultural—activities are internalized (Vygotsky, 1980). This process is nuanced as the social and cultural activities that surround a child are highly varied. Adults, be they parents, teachers, or members of the community, are gatekeepers of immense amounts and diverse types of information. Students arrive at school, for example, with sociocultural capital and their internalization of information is mediated by language and information (Bourdieu, 2011; Portes & Vadeboncoeur, 2003); children come to value and find differing meaning in activities by virtue of their early experiences.

Some of those early experiences fail to equip the student with the language necessary to work effectively in today's classrooms. It becomes the work of the teacher to engage the students in the discursive process of acquiring academic language (Gibbons, 2013). In Vygotskian terms, students' everyday concepts need to transition to academic concepts—a pedagogical process termed *metamessaging* by Forman and Larramendy-Joerns (1998). By using metamessages, teachers reword students' statements to align more appropriately with the terminology expected in the classroom. John-Steiner and Mahn (1996) warn that how and what students learn in out-of-school contexts and what they are taught within school directly influences school learning; therefore, children's early exposure to information is exceedingly important. Given that CS has only recently emerged as a potential academic topic, especially at the elementary level (Code.org & CSTA, 2018), it is possible that students' perceptions of this area of study and its associated language is likely to be very uneven and highly influenced by out-of-school exposure.

Given our focus on students' self-reports of their varying interests and experiences, our research objective is to develop a survey instrument appropriate for diverse upper elementary students that measures their attitudes and perspectives on CS.

Data and Methods

Cognitive Interviewing Process

Our cognitive interview process followed Karabenick et al.'s (2007) interview probes (see Table 1). During the interview, students were asked what the item meant, which answer they would select (from *strongly disagree* to *strongly agree* on a 5-point Likert scale), why that answer made sense for them, and other relevant probes (e.g., "What is engineering?"). The interviewer was encouraged to ask other germane questions emergent from the talk that would help the student express his or her understanding of the item. The students were interviewed individually by trained graduate researchers.

Table 1

Cognitive Interviewing Probes

Standard Interviewer Probes
"Please read item number … out loud to me."
"What does that mean?" or "What is that item asking you?"
"From these [Likert] responses, which would you pick as your answer?"
"Can you explain why you picked that answer?"

This protocol was used in three separate, iteratively-linked studies used to both garner a better grasp of students' understanding of key computer science terms but also develop a set of refined attitudinal survey items that displays a higher degree of stability of interpretation (gamma stability; Broderson & Thornton, 2011) across students of this age range. Below are the findings from these three studies.

Analysis

To assess students' understanding of individual items, two coders rated the students' responses to the items and interviewer prompts. Students' responses were listed verbatim and by item on a series of spreadsheets. In this way, the coders could utilize Karabenick et al.'s (2007) scoring methodology for assessing overall cognitive validity and the student responses were scored on a Likert-scale from 0 to 4, indicating the coders' assessment of the student's level of understanding of the individual item, from *clear evidence of misunderstanding* at 0 to *clear evidence of comprehensive understanding* at 4. Cognitive validity indicates the student's conceptual understanding of the item as determined by alignment in their verbal interpretation of what the item means, their explanation for why they selected the answer they did, and the compatibility of their Likert response with their explanation. The coders trained on a sample set of data, then independently completed their ratings. Finally, where there was disagreement on the cognitive validity scores of the items, the coders discussed and reached consensus. Items that students struggled to understand—as determined by the coders—were examined more closely for later modification in subsequent studies. Additional details regarding analyses completed within each study appear below.

Study 1

Participants

Our participants were 33 upper elementary students (ages 9-11) in two different schools in the southeastern United States. Table 2 presents demographic data on the schools and students that participated in our study. The school names provided are pseudonyms to preserve anonymity. Atwell School is a rural school with roughly equivalent percentages of African American, Caucasian, and Hispanic/Latinx students. Ellis School is an urban charter school with

80% Caucasian and roughly 5% each of African American and Hispanic/Latinx students. All interviewed students also participated in a large CS educational intervention study. In that study, the students were participants in a seven-day computer science elective that implemented a coding curriculum that the researchers designed. Students in Atwell were taught by the researchers whereas students in Ellis were taught by their technology teacher. There were 16 students interviewed at Atwell, 8 of whom were girls; 17 students were interviewed at Ellis, 9 of whom were girls.

Table 2

Study 1 School-level Demographics

School	Black	White	Latinx	Other	NSLP
Atwell	36%	29%	33%	2%	98%
Ellis	6%	80%	5%	9%	6%

Note. NSLP denotes students eligible for Free and Reduced Lunch

Methods

For our initial survey items, we modified items from a validated STEM attitudes instrument (S-STEM; Unfried et al., 2015) to create a CS version for upper elementary students. More specifically, the nine original items from the Technology and Engineering Attitudes sub-scale were used. These items covered two psychological constructs: self-efficacy and outcome expectancy (Wigfield & Eccles, 2000). For example, the original item “I like to imagine making new products” was modified to “I like to imagine making new computer programs” for Study 1. In Study 1, graduate students read all of the items to the elementary students, although the children were directed to follow along, and their responses to probes were transcribed verbatim. The questions from this study are listed in Table 3 below.

Table 3

Item Wording Changes

Original S-STEM Wording	Study One Wording	Final Study Three Wording
1. I like to imagine creating new products	1. I like to imagine making new computer programs	1. I would like to use coding to make something new
2. If I learn engineering, then I can improve things that people use every day	2. If I learn coding, then I can improve things that people use every day	2. If I learn coding, then I can improve things that people use every day
3. I am good at building and fixing things	3. I am good at building and fixing computer program	3. I am good at building code
4. I am interested in what makes machines work	4. I am interested in what makes computer programs work	4. I am good at fixing code
5. Designing products or structures will be important for my future work	5. Designing computer programs will be important in my future jobs	5. I am interested in what makes computer programs work
6. I am curious about how electronics work	6. I am curious about how computer programs work	6. Using code will be important in my future jobs
7. I would like to use creativity and innovation in my future work	7. I want to be creative in my future jobs	7. I want to use coding to be more creative in my future jobs
8. Knowing how to use math and science together will allow me to invent useful things	8. Knowing how to use math and science will help me to create useful computer programs	8. Knowing how to code computer programs will help me in math
9. I believe I can be successful in a career in engineering	9. I believe I can be successful in computer science and programming	9. Knowing how to code computer programs will help me in engineering
		10. Knowing how to code computer programs will help me in science
		11. I believe I can be successful in coding

Analysis

Three members of the research team engaged in thematic analysis (Braun & Clarke, 2006). These members represent expertise in psychology, education, and computer science. The purpose of the thematic analysis was to determine the themes that emerged from the students' responses, in particular if students emphasized certain experiences or concepts or introduced an example as a way of describing their perspective. After the interviews were transcribed, initial codes were determined, which were collapsed into the themes noted below. This inductive process privileged students' perspectives and experiences, which drove changes in survey item wording. These themes were agreed upon by consensus.

Findings

We conducted specific cognitive interviews on items 1, 8, and 9. These items were selected based upon our need to determine students' ability to understand computer programs as used initially in Item 1, but also used in the majority of the remaining items. If the students struggled to comprehend the concept in this item we surmised it would be problematic in other items and we resolved to consider alternatives.

Regarding Item 8, we wished to ascertain to what extent students might consider STEM-based courses as being supportive of one another. There is increased interest in developing strategies for computational thinking (CT) integration into STEM subject areas (National Science Foundation, 2019) and this policy interest is emerging in parallel with increased researcher interest (e.g., Sengupta et al., 2013; Weintrop et al., 2016). However, it is unclear whether students are aware of these CS/CT and STEM connections. Thus, we need questions that specifically probe for this. Moreover, we reversed the wording of the items; originally the student would have been primed to consider science or math first and then computer science second. Our concern was that students might associate needing refined skills in those subject areas in order to do well in computer science, so by privileging computer science students may consider how what they are currently doing may benefit their work in traditional classroom subjects.

Item 9 was of particular interest as it was intended to assess self-efficacy and it utilized the specific phrase computer science; we were interested to learn if elementary students understood this term. The purpose of the interview during Study 1 was to glean students' interpretation of the items. In other words, we needed to know to what extent we and the students had the same understanding of these terms.

Item 1: I like to imagine making new computer programs

Themes for this item included *Computer Games/Usage* and *Coding Experience*. Of the 33 students interviewed, eight responded to the probe "What are computer programs?" in Item 1 by noting games or apps, either directly (e.g., Facebook) or generally, "a program is something you play or do on a computer." Seven students shared coding examples or experiences. These responses included, "creations that people can create through coding" and "a series of code, strands of code that when put into a computer, the computer does it." Of the remaining students, seven answered "I don't know," five were able to supply a response close to our intentions (i.e., "things that a computer is told to do"), and the rest gave one-off answers that we could not easily categorize (i.e., "documentaries" and "digital clubs"). Because many of the students supplied the word *coding* in their responses, the team shifted phrasing as noted in Table 3. The logic behind the rewording was that students might be prompted to connect coding with the creation of computer programs by seeing the phrase *coding new computer programs*.

Item 3: I am good at building and fixing computer programs

Item 3 probed students on their self-concept of ability in building and fixing code. Students in this study were only asked to supply their answer—*strongly disagree* through *strongly agree*—to this item. Of the 33 students queried, 4 students freely offered that they had different answers for each action. These students considered them distinct processes or skills, thus, we opted to split this item into two as a result.

Item 8: Knowing how to use math and science will help me to create useful computer programs

Student responses to this item fell into two themes: *Reiteration* and *Personal Experience*. 42% of students simply restated the item or responded in general terms (i.e., “the basics of math and science will help me learn more”) when asked if math and science would help them create useful computer programs. Seven students provided examples of the ways they experienced, or could imagine, math helping in coding. Student responses included, “Math are the variables, science.... I don’t know.” Only two expressed their connections to science as “technology” and “physics.” To this end, the team determined to split the single item into different items with math, engineering, and science. In this way, additional probes could clarify students’ understanding of the relationship of these subjects to CS.

Item 9: I believe I can be successful in computer science and programming.

Themes for this item fell into several major categories: *Unsure*, *Coding*, *Computer Operation*, and *Science*. When prompted to answer, “What is computer science?” students were fairly evenly distributed; 14 answered “I don’t know” whereas 10 answered by making programming or coding connections. The remaining students made general statements about how computers work (i.e., “what makes a computer work”) or statements focused on the word science (i.e., “science on computers, different science, not normal science”). The team shifted wording on all items to only include coding. Our thinking was that this term reflected processes the students most likely encountered, though Hour of Code activities, for example.

Discussion

In summary, students struggled to understand *computer programs* and *computer science*, often confusing these concepts with general computer usage and with specific applications like Facebook and with computer games. As such, we determined that *coding* captured the essence of our interests and that young students were more likely to be familiar with, and have an understanding of, this word. Furthermore, our initial results indicated that approximately 75% of students did not understand how math and science together were connected to coding. We therefore made the decision to split the single item into three different items (math, science, and engineering). This was done to reduce extraneous cognitive load (Sweller, 1988) as students would have to consider math and science individually, then together, and finally to consider how they may foster their use of coding. The team added the third term—engineering—as a way to assess to what extent young students understood what engineering is and how processes involved in this practice might align with computer science. Lastly, despite the fact that only a few students expressed that building and fixing code were distinct skills, we split the item into two. Our goal in doing so was twofold; one, to reduce cognitive load as noted above; and two, to elicit more detailed information from students by probing on the nuances between building code and debugging it. The changes to the items that we made following Study 1 appear in Table 3. We must note that we dropped an item—I am curious about how computer programs work—from Study 1, having found that students repeatedly asked the interviewer "didn't I already answer that?" The wording was very similar between two items (Items 4 and 6), so we opted to retain the one that appeared to be less confounding for students.

Study 2

Participants

Our participants were 31 upper elementary students (ages 9-11) in two different schools in the southeastern United States. Table 4 presents demographic data on the schools and students that participated. The school names provided are pseudonyms to preserve anonymity. We returned to Atwell School to conduct cognitive interviews. Again, Atwell is a rural school with roughly equivalent percentages of African-American, Caucasian, and Hispanic/Latinx students. Franklin School is a suburban school with over 50% Caucasian and approximately 20% each of African-American and Hispanic/Latinx students. Students in Study 2 did not participate in any CS-specific intervention and the interviewed students at Atwell did not participate in our previous intervention nor in earlier interviews. It was important for the development of appropriately worded items for us to query students from diverse socio-demographic backgrounds. There were 22 students interviewed at Franklin, 7 of whom were girls; 11 students were interviewed at Atwell, 6 of whom were girls.

Table 4

Study 2 School-level Demographics

School	Black	White	Latinx	Other	NSLP
Franklin	18%	55%	22%	5%	36%
Atwell	36%	29%	33%	2%	98%

Note: NSLP denotes students eligible for Free and Reduced Lunch.

Methods

In Study 2, the students read the items aloud and the entire interview was audio recorded and transcribed. The interviews took approximately eight to 15 minutes. Over the course of the interviews, the students were asked if any words were confusing or if they did not know the meaning of a word or phrase. As in Study 1, due to time constraints in the classroom, a subset of items were chosen for cognitive interviews. These items were chosen based on changes and findings from Study 1. These will be discussed in detail below. The purposes of the interviews shifted from Study 1; here we were most interested in asking the students why they selected the answers they did. More specifically, we wanted to see in what ways students' chosen Likert-responses aligned with or deviated from their open responses.

Analysis

Similar to Study 1, after all the interviews were completed and transcribed verbatim, we thematically analyzed students' responses (Braun & Clarke, 2006) for five items. The first author generated initial codes by reading through students' responses and pulling out salient phrases or words. Other authors then peer checked (Lincoln & Guba, 1985) the initial codes and combined them to form themes. These themes were continuously checked against the data and to scan for patterns. Consensus was reached for all coding decisions. Additionally, two members of the research team assessed the students' responses for cognitive validity, rating the students' overall understanding of the item on a 0 to 4 Likert scale. Given that the ratings should be considered ordinal and not continuous, we calculated polychoric correlations (Nunnally, 1978) to assess rater agreement. The initial level of agreement ranged from 0.53 to 0.82. For items for which there was disagreement consensus was used to reach agreement.

Findings

Item 1: I like to imagine coding new computer programs

The student responses to this item broadly fell into four themes according to the word or phrase on which students focused. Those who privileged *liking* responded by noting, "I just like to play around" and "I don't like technology like that." Other students highlighted the phrase *to* *imagine* in their responses by stating, “I don’t imagine about computers... [imagine means] fake things and ... different things that might happen in the future.” A third group of students focused on *coding* and noted, “Because I feel like when you’re coding, you’re in a whole different world and like you are in charge of whatever you’re coding and you do what you want to do,” and “Because, ... I don’t think I’m going to be a computer programmer, a coding person.” A final group—*computer programs*—spoke of “not really [being] big on computer programming and stuff. I’m more an outdoors person” and “I like to imagine coding new computer programs because new computer programs can help people... like [they] can make things easier to access.”

Items 3 and 4: I am good at building/fixing code

Students were asked to respond to these probes as individual items; however, their replies together highlight intriguing understandings of how elementary students consider the process of working with code. One student, for example, noted that she somewhat disagreed with being good at building code because “if I had to code by myself I’d probably get halfway through and just stop because I’m too impatient;” however, she stated she somewhat agreed with being good at fixing code because, “when someone messes up in code, I can fix it.” Another student selected strongly agree for building code noting “we had to make a game... and I had to find out how to make all of it.” He chose neither agree nor disagree for fixing code and explained his selection by stating, “some of the time I can fix the code... [and] and other times, I keep looking and... I can’t fix it.” Students from Atwell School selected the disagree options more often (approximately 64%) than the Franklin School students (approximately 18%). Many of the Atwell students shared that they did not know what building nor fixing code meant and one noted that their teacher “only taught doing it, not like fixing it.” Moreover, of the 31 students queried on these split items, 15 offered different answers for each.

Item 9: Knowing how to code computer programs will help me in engineering

The most noticeable deviation in student responses to this item occurred between schools. Franklin School—with its weekly digital technology class for every homeroom—had student explanations such as "I feel like engineering has a lot to do with technology. And these days coding has a lot to do with technology as well" and "Because engineering is almost the same thing as coding. Except engineering is making the actual thing and coding is telling... that thing [what] to do." Atwell School—where there was more limited access to computers and technology-related activities—had student responses such as "I don't do engineering. We haven't learned that yet" and "In engineering, you work on cars." All students in Study 2 were asked to provide a definition for engineering. Franklin School students' definitions included: inventing and building, robots and technology, and building cars. Over 50% of Atwell students offered a definition that included "fixing cars."

Item 10: Knowing how to code computer programs will help me in science

Students' responses to this item fell into three major themes: *science is technology, science is hands-on,* and *general computer use*. More specifically, *science is technology* students explained the following, "Science involves a lot of technology... It's like math, it's like a lot of different things combined like technology and math" and "Scientists have to code robots and that coding... means it's telling it what to do. Also, there's different types of sciences [and] one of those types do stuff with electronics." *Science is hands-on* responses included, "I mean science is like real life stuff and how to make chemical reactions and stuff like that. And coding is how to work with computers and make them work" and "Hm, 'cause I think of science as like putting different stuff together and testing things and learning about rocks and minerals and stuff. I wouldn't really use coding for science. I'd rather mix stuff together, make new things, and go outside and study." *General computer use* responses included, "Coding can help people when it comes to learning how to do stuff, when it comes to being online."

Discussion

For item 1 (I like to imagine coding new computer programs) students did not focus on the terminology and intent we expected. We concluded that the wording of the item was unclear; students interpreted it in one of several ways rather than as a singular probe of whether they envisioned themselves using code to develop something new. We did not anticipate the word "imagine" would be problematic for the students as we conceptualized it to be an expression of interest. This highlighted the uncentered wording of the item and perhaps a vocabulary disconnect. Beers (2003) suggests that active and effective readers use context clues and their prior knowledge to monitor their understanding of a text. Students concentrated on the concepts on which they could pull from experience or offer an appropriate response. The focus of some students on "coding" and others on "computer programs" is also important to note. Further investigation is required to understand whether the students in either category are able to connect both of those terms and whether they understand each of these terms thoroughly. Because of the overall finding of uncentered wording, the research team opted to reword the item entirely.

Modifying the original item 3 from a singular probe to two distinct items in Study 2 was appropriate as approximately half of the students queried made distinctions between their self-concept of ability in building versus fixing code. Our findings are consistent with previous work which concluded that debugging is a different set of skills than programming (Ahmadzadeh et al., 2005; Brennan & Resnick, 2012; Tran, 2019). Our finding highlights the fact that students' actual and perceived abilities in this area should be considered separately. As such, the research team felt comfortable with the wording of these items and keeping them as distinct probes.

Item 9 appears to highlight disparities in schema development (McVee et al., 2005). Overall, one group of students had a more robust understanding of engineering as a practice and could better conceptualize of how coding and engineering are synergistic skill sets and future occupations. In alignment with sociocultural theory, students arrive at school with widely varying home and community experiences and then receive additional distinct opportunities once there. Thus, schools and communities provide students access to both common and unique sources of culturally valued information, artifacts, and resources (Archer et al., 2015; Portes & Vadeboncouer, 2003; Vygotsky, 1980). Disparities may occur due to unequal access to the cultural and technological resources because of inequitable funding structures or geographic limitations. Children bring both differing vocabularies and understandings of words to school. Our results highlight this; students from the suburban school (Franklin), in what is considered a "high-tech" employment region, had more developed understandings and experiences from which to draw and respond, whereas students from Atwell—situated in an under-resourced rural community—likely had fewer such experiences and therefore were unable to make the relevant connections to engineering as a diverse field of study.

Item 10 illustrates both the sociocultural context at play and a disconnect of educational policy and practice from public understanding. Science is an expansive domain, with unclear boundaries, and often is misrepresented and misunderstood (Feinstein, 2015). Efforts to bring science into the public sphere—to make it interdisciplinary, more accessible, or immediately relevant—include the use of engaging and timely socioscientific issues (SSI; Zeidler, 2014) in classroom instruction. Student responses appear to support the use of such pedagogical approaches in conjunction with a computer science curriculum in order to help them move beyond stereotypical or superficial interdisciplinary connections between science and coding.

In general, the term *coding* appeared multiple times throughout students' responses when they are talking about CS. Students' past experiences in both formal and informal environments can contribute to the development of a conceptual understanding (Pines & West, 1986). The fact that students' understanding of computer science often revolved around how or if they could relate coding to CS, would be an outcome of previous experience in coding. Fundamental terms in computer science like this can thus be utilized in formal instructions to help students with CS concept development and domain identification. The research team felt comfortable with the wording of items 9 and 10 as the students were able to respond to the probes appropriately. However, we wished to gather more qualitative data on how students perceived the connections between CS and these other subject areas. Our hope was to begin to outline pedagogical implications based on students' experiences. To this end, we continued to query students on these items.

Study 3

Participants

Our participants were 32 elementary students (ages 8-11) in two different contexts in the southeastern United States. Table 5 presents demographic data on the contexts and students that participated. The first context was a summer camp associated with the university and intended for rising 3rd through 5th grade students. The second context was a suburban school called Harris. Harris' student population is approximately 50% Caucasian, 20% African-American, and 15% Hispanic/Latinx. Student participants at Harris were in 5th grade and were pulled from their computer science class to participate in the interviews. The combination of the contexts and return rate of consent forms resulted in this study being disproportionately male. There were 18 students interviewed at the summer camp, 5 of whom were girls; 14 students were interviewed at Harris, 6 of whom were girls.

Table 5

Study 3 Demographics

Context	Black	White	Latinx	Other	NSLP
Summer Camp	31%	30%	13%	25%	N/A
Harris	20%	54%	16%	9%	28%

Note: NSLP denotes students eligible for Free and Reduced Lunch. Summer Camp demographics are for the study participants and not the entire camp

Methods

In Study 3, the students read the items aloud and the entire interview was audio recorded and transcribed. The interviews took between five and 15 minutes. Over the course of the interviews, the students were asked if any words were confusing or if they did not know the meaning of a word or phrase; one interview was ended early by the interviewer as the child was unable to describe “coding.” As in the earlier studies, due to time constraints in the classroom and camp activities, a subset of items were chosen for cognitive interviews. These items were chosen based on changes and findings from Study 2. These will be discussed in detail below. The purposes of the interviews shifted from Study 2; in the current study we were most interested in eliciting from students how they understood the wording of the item as well as why they selected the answers they did.

Analysis

Mirroring analyses from Study 1 and 2, once the interviews were transcribed verbatim, the first two authors generated initial codes and collapsed them into themes, which were peer checked by others on the research team. Consensus was reached for all coding decisions. This thematic analysis occurred for three items. Similar to Study 2, we calculated polychoric correlations (Nunnally, 1978) to assess rater agreement. The initial level of agreement ranged from 0.54 to 0.80. For items for which there was disagreement consensus was used to reach agreement.

Findings

Item 1: I like to use coding to make something new.

The majority of the responses fell into the theme of *Item Reiteration* (N=18). Responses that fell under this category include, “If you would use coding and make something new out of it…” Under *Item Reiteration,* we separated responses under the subcategories *Close Reiteration* (N=9) and *New* (N=15). Those in the second subcategory contained sentences that focused on the word “new.” An example of an answer that contained both is, “It is asking me to use different types of coding to make something completely new out of that coding. Like making a new program that can answer something that no one else has really answered.” The first sentence in the response is a *Close Reiteration* and the second is a deeper explanation that focuses on “new.”

Two other themes that emerged from the students’ answers related to this item were about the *Goal of Coding* (N=9) and the students’ *Attitudes Towards Coding* (N=2). An example of a response containing goals of coding is, “To make some kind of new program. So programming a game or a… or something where kids can use that coding to learn how to code in Scratch.” The attitudes towards coding includes, “That you like coding” and “It means that if you really want to make something, like a new invention or something, you can use coding to do it and that would be pretty cool.”

The remaining responses were unlike those noted above. Some of them contained examples of specific blocks (e.g., *move* blocks) that the students likely used in classes. Other interesting responses were, "That um, I wanna like, try new things and [do them] through coding" and "It means using coding, you can create something and that is... new in the sense that it's coming from you…" We found these responses particularly interesting because the students were focused on trying new things, and the new things were their creation. The second quotation implies that the student feels ownership over his/her creation.

Item 9: Knowing how to code computer programs will help me in engineering

Students' responses to this item fell into two large themes—*Career* and *Item Reiteration*. Career-based responses (N=11) included mentions such as, "It's asking if coding, if you want to be an engineer, it will help you with your job" and "Like [coding] will help you when you're engineering stuff if you become an engineer." Item reiteration responses (N=14) were simply instances in which students restated the item but did so by substituting their own words and/or by providing explanatory examples. Such examples include "I think it's asking me, like, how… coding computer programs help you in like, like, engineering … like how to build stuff..." and "It is asking me if knowing how to code the computer, create programs, or games, or websites, would help me in engineering when I'm building something."

Student responses also resulted in other themes, fewer in number, but worthy of note. These included *Math*; *Coding in Engineering*; and *Cars, Robots, and Technology*. The math-based responses included this statement: "Well code can help you learn math because you have to be able to use math to code sometimes. And in engineering you have to use math." *Coding in Engineering* responses included pronouncements such as, "How coding, if you know how to code you can know how to engineer stuff" and "It is asking if computer programming will help in engineering for people, and how it will maybe benefit them." The last theme—*Cars, Robots, and Technology*—is an aggregate of examples students offered for how they see coding directly applying to their understanding of engineering. For example, "So if you're engineering, ... you would need um code- coding 'cause sometimes you can make like any type of technology maybe. Anybody can make like a hovering car, like a real-life hoverboard. Like you would need coding for that."

Item 10: Knowing how to code computer programs will help me in science

Students' responses to item 10 fell into three themes, although the vast majority of students (N=22) reiterated the item by rewording it and/or providing an example. Such *Item Reiteration* responses include, "It's asking me that in science, knowing how to code will be helpful" and "How code can help you in science… and how you can use computer programs to help you understand what science is better." Of note, only two students mentioned *Careers* in their response to this item: "It's asking you if coding computer programs will do something if I like to do science… Because I am going to have to use a lot of science when I grow up because I'm going to have multiple jobs, I think." The final theme is an aggregate of how students *Connect Science to Coding*. Three students explicitly noted that they do not see how coding and science connect. One such response was "Because [in] science you learn about volcanoes and how they work and I don't think coding really involves science." The remaining responses in this theme show alignment between science and coding, albeit in varied ways. For example, "Like, if you know how to code a computer, how it will help you in science… Because, um, it has a little bit of science to it, and it also has like experimenting in it to see what happens, and that's kind of part of science."

Discussion

For Item 1, over half the students queried offered item-reiteration responses. The majority of these responses focused on the word "new" in the item. Because of the number of students that seemed to understand the intent of the item, we believe that the wording of the item is appropriate for upper elementary students. It is also important to note that under *Goal of Coding*, the majority of the responses (N=7) contained sentences about games. The students that spoke about games seemed to have trouble coming up with any other specific examples of new programs they can create, "Like, to use coding, coding is a computer thing to use on a computer to make a game or any, to make characters move or to create something." This reflects the types of activities that students associate with coding. In order to better help students understand that they can complete a wide variety of tasks by coding, practitioners and researchers should focus on curricula and activities that are more reflective of problems computer scientists solve.

Regarding item 9, just under half the students queried offered item-reiteration responses. However, many of the career-themed responses also restated the item in such a way that students clearly understood the intent of the item as we hoped. As such, we feel confident moving forward with this item as worded. It is important to note that this item, more so than item 10 below, had more career-themed responses. We posit that elementary students do not typically take classes in engineering as they do in science or math and may not conceptualize it as being anything other than an activity that occurs distally, as an adult.

Students' responses to item 10 largely support our intent in writing the item. The students were able to connect coding and science in ways that highlight how the skills learned through coding could help their learning in science. It is important to note that students' understanding of science is varied; they offered definitions that ranged from "the learning of everything" and "think[ing] of new stuff and new ways to help people" to the more specific topics within science such as chemistry, planetary studies, force and motion, and the human body.

Across all three items, several responses are worth highlighting. One student suggested that coding seemed more appropriate in an English language arts class: "[It's] like learning new words or fixing things I guess. Let's say I was writing a summary and I wanted to fix some things, and that is technically coding because I am fixing things that I messed up."

Another student conflated cryptography and coding, noting at one point and in response to item 1, "I guess to like, uncode an answer, and to use different codes to make like words. Or you can do different things with codes." In response to item 9, this student offered this: "It is asking me to figure out a way to like engineer different, like, machines that can take in codes and decode them or you can code one then decode it and code it again." We fully recognize that cryptography is an important topic under cybersecurity within computer science; that a student has conflated these concepts underscores the need for the CS education community to more fully utilize appropriate terminology within CS activities for young students.

Moreover, two students expressed the benefits of using coding as a planning and modeling tool to help with the *doing* of science or engineering. One noted, "Engineering is one of those things where you need the 3D model- modeling and you've got a lot of math involved in that." The other stated, "Well engineering is kind of like designing things and making things better, even just making a new invention. And you can use coding to make it work on there before you actually start it because if you actually start it and you don't use coding then it will be hard to plan it out, I guess. And if you don't use coding or a plan to start it before you actually do it then if you make mistakes you can't fix it on the coding."

Final Discussion

Cognitive interviews are a potent tool for systematically investigating children's self-reports with an acceptable cognitive validity (Woolley et al., 2004). In the process of developing this instrument, we analyzed multiple aspects of the students' responses. These include the students' self-concepts regarding CS problem solving, their understanding of CS as a domain, perceptions of other related domains, their understanding of CS specific concepts like fixing/debugging, and the associated prompts as well (i.e., "what is engineering?"). Overall, our findings reinforce the importance of revalidating instruments when adapted to new foci or used with younger audiences. If we had simply taken the S-STEM instrument, designed for middle grades students with an engineering & technology focus, and adopted it for elementary students with a CS focus without this cognitive interviewing process, we would have likely had psychometrically problematic results. As an added benefit, our studies provided important insights into children's thinking around core CS concepts, thus informing both curriculum development and pedagogical strategies.

Our findings support our view that prior experience and opportunities afforded to students shaped their responses as much as their general developmental level. Piagetian theory supports that children in the 9 to 11 age range can think about and solve problems that pertain to real, or actual, objects (Piaget, 2002). This may well be why we saw some marked differences in responses between the schools. Some students had actual experience with concepts about which they were probed, whereas others had no such experiences. This lack of experience would have been too abstract for this developmental phase. One immediate implication of this is the need to imbue elementary curricula with terminology and experiences that connect with and transcend what the children encounter in their own communities. Our findings are in alignment with both sociocultural theory in general (Portes & Vadeboncouer, 2003; Vygotsky, 1980), and social capital theory as applied to STEM areas (Archer et al., 2015). At the policy level, our findings reinforce stated concerns about the opportunity gap for youth with regards to exposure and cultivating interest in high-value STEM career pathways (Code.org & CSTA, 2018).

It is important to reconsider and reword survey items when exploring specific domains. Our expert conceptions of appropriate language do not always correspond with young students' understandings of terminology. For example, over the course of these studies, item 1 underwent several important changes. The original S-STEM (Unfried et al., 2015) Technology and Engineering wording was "I like to imagine making new products." The team considered "I like to imagine making new computer products" in order to adhere as closely as possible to the original. This was rejected in favor of "I like to imagine making new computer programs." Still, students had difficulty with this wording as they could not isolate what computer programs were. In hopes of probing their thinking, we shifted wording slightly to "I like to imagine coding new computer programs." Students then grappled unnecessarily with this item as they did not focus on what we hoped: coding for creation. As such, we shifted wording to "I would like to use coding to make something new."

The language educators use with students is exceedingly powerful. Helping students broaden the connections between their everyday and scientific language may prove influential. One such example is in our use of "fixing" as a synonym for "debugging." One student offered the following explanation of fixing: "I can code and I think that fixing code is probably a little harder because it includes understanding the code and then being able to change it and make it better, or fix something that is wrong with it." Students did not often share such profound understandings of fixing/debugging. Care needs to be taken to ensure that students' comprehension of essential CS concepts straddles not only diverse socio-demographic school contexts but also from primary to intermediate to secondary education levels. Some students' domain understanding of subjects such as engineering and science—and the processes involved in these subjects—are still blurry to elementary students. Because these terms are so interconnected as well as important to 21st century learning, curriculum writers and educators need to focus on how to make the domain-specific terminology clear.

Limitations

Findings of these studies should be considered in the context of the following limitations. The first is sample size and methodology. Cognitive interviewing is time intensive in nature; therefore, we opted to prioritize quality over quantity. As such, we purposefully selected contexts that reflected a diverse array of student backgrounds so as to capture a range of student responses and experiences. The second limitation is our decision to probe students on only a select number of items. This likely could have been remedied by conducting the studies in a lab setting; however, we chose natural learning environments in order to open the study to as many possible students. The third limitation is in reference to the socio-demographic variables we were able to collect; future work might consider collecting more variables. Lastly, some of the student participants expressed that they had never been interviewed and others shared that they had never been asked questions about their understanding of a statement. The cognitive interviewing process was new for all the students and may have caused some discomfort for some who felt they needed to share a *right* answer with the researchers.

Conclusions and Future Work

Over the course of these three studies, we qualitatively analyzed students' responses to the wording of thirteen items, resulting in a final set of eleven items deemed appropriate for upper elementary students. We intend to distribute our survey in order to quantitatively validate the instrument. As such, we anticipate this instrument being used to measure 4th and 5th grade students' attitudes toward CS, in particular before and after completing a relevant intervention.

There is still much to understand about how elementary students learn computer science. Because their perceptions of CS can affect their learning, it is important for researchers in the computer science education community to study what those perceptions are, how they came to be that way, and the importance of addressing any misconceptions in a manner that can broaden participation in the field. Our results contribute to the CS literature by showing the varied ranges of conceptions young students have regarding these concepts. As can be seen from their responses to probes of their thinking, upper elementary students often have somewhat I and vague understandings of core and essential vocabulary such as computer science and programming. Additionally, young students have diverse notions of how CS may connect with other subjects they learn. Our results are also of pragmatic curricular interest as they highlight voids in instruction regarding not just key vocabulary, but perhaps of specific CS processes like debugging.

Students' varied conceptions of how CS/CT and other STEM subjects might align is in agreement with prior work on student conceptions of how STEM academic areas relate to each other and to future career pathways (Wiebe et al., 2018). These findings point to any number of potential interventions to be implemented and studied. Of particular interest would be if students' awareness changes after participating in a researcher led intervention specifically designed around CS/CT integration into traditional STEM academics (e.g., science and mathematics).

The computer science education community may consider taking up research using this systematic approach to cognitive interviewing to ensure measures of young students' attitudes and perspectives are cognitively valid. There are several potential directions to consider. For example, we may be interested in knowing to what extent the students self-regulate their learning—how and if they plan, monitor, and evaluate—in a coding environment. Moreover, including an open-ended prompt, allowing students to offer any final thoughts, may elicit rich information about which we never thought to inquire. Finally, querying students on their attitudes toward collaborative coding may indicate potential roadblocks and solutions to encourage students to work together on coding activities. Cognitive interviewing is a powerful tool for researchers interested in developing more stable and reliable instruments. Moreover, curriculum developers, who wish to create materials that provide students with authentic learning experiences that help to bridge their existing understanding with new content, may find it helpful as well. As seen from these three studies, it was an important tool to capture students' understanding of CS language and processes.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No DRL 1721160. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We also thank the teachers and camp counselors who provided us time and space to conduct our interviews and the students who devoted their time and energy to provide thoughtful responses to our questions.

References

Ahmadzadeh, M., Elliman, D., & Higgins, C. (2005, June). An analysis of patterns of debugging among novice computer science students. In *Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education* (pp. 84-88).

Ajzen, I., & Gilbert Cote, N. (2008). Attitudes and the prediction of behavior. In W. D. Crano & R. Prislin (Eds.), *Attitudes and attitude change* (pp. 289–311). New York: Psychology Press.

Alano, J. et al. (2018). *K12 Computer Science Framework*. Retrieved from: https://k12cs.org/

Archer, L., Dawson, E., DeWitt, J., Seakins, A., & Wong, B. (2015). "Science capital": A conceptual, methodological, and empirical argument for extending bourdieusian notions of capital beyond the arts. *Journal of Research in Science Teaching, 52*(7), 922-948.

Arthur, M. W., Hawkins, J. D., Pollard, J. A., Catalano, R. F., & Baglioni Jr, A. J. (2002). Measuring risk and protective factors for use, delinquency, and other adolescent problem behaviors: The Communities That Care Youth Survey. *Evaluation Review*, *26*(6), 575-601.

Baars, R. M., Atherton, C. I., Koopman, H. M., Bullinger, M., & Power, M. (2005). The European DISABKIDS project: development of seven condition-specific modules to measure health related quality of life in children and adolescents. *Health and Quality of Life Outcomes*, *3*(1), 70.

Baker, R. S., D'Mello, S. K., Rodrigo, M. M. T., & Graesser, A. C. (2010). Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies, 68*(4), 223-241.

Beers, K. (2003). *When kids can't read: What teachers can do*. Portsmouth, NH: Heinemann.

Bourdieu, P. (2011). The forms of capital (1986). In I. Szeman & T. Kaposy (Eds.), *Cultural theory: An anthology* (pp. 241-258). Chichester, UK: John Wiley

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77-101.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vancouver, Canada (Vol. 1, p. 25)

Brodersen, D. A., & Thornton, G. C. (2011). An investigation of alpha, beta, and gamma change in developmental assessment center participants. *Performance Improvement Quarterly, 24*(2), 25-48.

Carter, L. (2006). Why students with an apparent aptitude for computer science don't choose to major in computer science. *ACM SIGCSE Bulletin, 38*(1), 27-31.

Code.org, & CSTA, C. S. T. A. (2018). *2018 State of Computer Science Education: Code.org*. Retrieved October, 2018 from https://advocacy.code.org/

Feinstein, N. W. (2015). Education, communication, and science in the public sphere. *Journal of Research in Science Teaching, 52*(2), 145-163.

Forman, E. A., & Larreamendy-Joerns, J. (1998). Making explicit the implicit: Classroom explanations and conversational implicatures. *Mind, Culture, and Activity, 5*(2), 105-113.

Gibbons, P. (2013). *Scaffolding language, scaffolding learning: Teaching ESL children in the mainstream classroom*. Portsmouth, NH: Heinemann.

Graesser, A. C., McDaniel, B., Chipman, P., Witherspoon, A., D'Mello, S., & Gholson, B. (2006). Detection of emotions during learning with AutoTutor. In *Proceedings of the 28th Annual Meeting of the Cognitive Science Society* (pp. 285-290).

Grover, S., Pea, R., & Cooper, S. (2014, March). Remedying misperceptions of computer science among middle school students. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (pp. 343-348). ACM.

Hewner, M. (2013, August). Undergraduate conceptions of the field of computer science. In *Proceedings of the Ninth Annual International ACM conference on International Computing Education Research* (pp. 107-114). ACM.

Hewner, M., & Guzdial, M. (2008, September). Attitudes about computing in postsecondary graduates. In *Proceedings of the Fourth International Workshop on Computing Education Research* (pp. 71-78). ACM.

John-Steiner, V., & Mahn, H. (1996). Sociocultural approaches to learning and development: A Vygotskian framework. *Educational Psychologist, 31*(3-4), 191-206.

Karabenick, S. A., Woolley, M. E., Friedel, J. M., Ammon, B. V., Blazevski, J., Bonney, C. R., ... & Kelly, K. L. (2007). Cognitive processing of self-report items in educational research: Do they think what we mean?. *Educational Psychologist, 42*(3), 139-151.

Kukul, V., Gökçearslan, S., & Günbatar, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. Eg˘itimTeknolo- jisi Kuram ve Uygulama / *Educational Technology-Theory and Practice, 7*(1), 158-179.

Lewis, C. M., Anderson, R. E., & Yasuhara, K. (2016, August). I Don't Code All Day: Fitting in Computer Science When the Stereotypes Don't Fit. In *Proceedings of the 2016 ACM*

Conference on International Computing Education Research (pp. 23-32). ACM.

Lincoln, Y. S. (1985). *Naturalistic inquiry*. The Blackwell Encyclopedia of Sociology.

McVee, M. B., Dunsmore, K., & Gavelek, J. R. (2005). Schema theory revisited. *Review of Educational Research, 75*(4), 531-566.

National Science Foundation. (2019). *STEM + Computing K-12 Education (STEM+C)*. Retrieved from https://www.nsf.gov/funding/pgm_summ.jsp?pims_id=505006

Nunnally, J. C. (1978). *Psychometric theory*. New York, NY: McGraw-Hill.

Özyurt, Ö., & Özyurt, H. (2015). A study for determining computer programming students' attitudes towards programming and their programming self-efficacy. *Journal of Theory and Practice, 11*(1), 51-67.

Piaget, J. (2002). *Judgement and reasoning in the child*. Routledge.

Pines, A. L., & West, L. H. (1986). Conceptual understanding and science learning: An interpretation of research within a sources-of-knowledge framework. *Science Education, 70*(5), 583-604.

Portes, P. & Vadeboncoeur, J. (2003). Mediation in cognitive socialization: The influence of socioeconomic status. In: A. Kozulin, B. Gindis, V. Ageyev, & S. Millier (Eds.), Vygotsky's *educational theory in cultural context* (pp. 371– 392). Cambridge: Cambridge University Press

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science, 45*(5), 583-602.

Reis, R. C. D., Isotani, S., Rodriguez, C. L., Lyra, K. T., Jaques, P. A., & Bittencourt, I. I. (2018). Affective states in computer-supported collaborative learning: Studying the past to drive the future. *Computers & Education*, *120*, 29-50.

Schaeffer, N. C., & Presser, S. (2003). The science of asking questions. *Annual Review of Sociology*, *29*(1), 65-88.

Schwarz, N. (1999). Self-reports: how the questions shape the answers. *American Psychologist, 54*(2), 93.

Schwarz, N. (2007). Attitude construction: Evaluation in context. *Social Cognition*, *25*(5), 638-656.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, *18*(2), 351-380.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, *12*(2), 257-285.

Tran, Y. (2019). Computational thinking equity in elementary classrooms: What third-grade students know and can do. *Journal of Educational Computing Research, 57*(1), 3-31.

Tsai, M. J., Wang, C. Y., & Hsu, P. F. (2019). Developing the computer programming self-efficacy scale for computer literacy education. *Journal of Educational Computing Research*, *56*(8), 1345-1360.

Unfried, A., Faber, M., Stanhope, D. S., & Wiebe, E. (2015). The development and validation of a measure of student attitudes toward science, technology, engineering, and math (S-STEM). *Journal of Psychoeducational Assessment, 33*(7), 622-639.

Vygotsky, L. S. (1980). *Mind in society: The development of higher psychological processes*.

Harvard university press.

Weintrop, D. (2016). *Modality matters: Understanding the effects of programming language representation in high school computer science classrooms*. (Doctoral dissertation). Retrieved from http://www.terpconnect.umd.edu/~weintrop/papers/WeintropDissertation.pdf

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147.

Wigfield, A., & Eccles, J. S. (2000). Expectancy–value theory of achievement motivation. *Contemporary Educational Psychology, 25*(1), 68-81.

Wiebe, E., Unfried, A., & Faber, M. (2018). Relationship of STEM attitudes and career interest. *Eurasia Journal of Mathematics, Science and Technology Education. 14*(10).

Woolley, M. E., Bowen, G. L., & Bowen, N. K. (2004). Cognitive pretesting and the developmental validity of child self-report instruments: Theory and applications. *Research on Social Work Practice, 14*(3), 191-200.

Zeidler, D. L. (2014). Socioscientific issues as a curriculum emphasis. Theory, research, and practice. In NG Lederman & SK Abell (Eds.), *Handbook of research on science education*, 2, 697-726.

CHAPTER 3: RELATIONSHIP BETWEEN RACE AND GENDER IN ELEMENTARY COMPUTER SCIENCE ATTITUDES: A VALIDATION AND CROSS-SECTIONAL STUDY

A version of this chapter appears as:

Vandenberg, J., Rachmatullah, A., Lynch, C., Boyer, K. E., & Wiebe, E. (forthcoming). Interaction effects of race and gender in elementary computer science attitudes: A validation and cross-sectional study.

Abstract

Computer science (CS) initiatives for elementary students, including brief Hour of Code activities and longer in- and after-school programs that emphasize robotics and coding, have continued to increase in popularity. Many of these initiatives are intended to increase CS exposure to students who historically have been underrepresented in CS academic trajectories, including women and students of color. This study aimed at examining the gender and race difference in elementary students' attitudes towards CS. To that end we developed and validated a survey instrument called Elementary Computer Science Attitudes (E-CSA) which consisted of the constructs of CS self-efficacy and outcome expectancy, through a combination of classical test theory and item response theory. The target audience for this instrument and study was upper elementary students (grades 4 and 5, ages 8 to 11). The E-CSA was found to be a gender and race bias-free instrument. A two-way ANOVA test was then used to answer research questions. We found no significant interaction effect between gender and race in the two constructs of CS Attitudes. We also did not see a significant difference based on race. However, a significant difference was found in both CS attitudes constructs based on gender, whereby male students had higher CS attitudes than female students. We discuss our findings from the perspective of the

equity issue in CS education. Furthermore, we believe the E-CSA instrument can inform classroom-based interventions, the development of curricular materials, and reinforce findings from cross-sectional CS studies.

Introduction

Despite a need for computing knowledge for educational purposes and career advancement, there remain challenges to attracting students to computationally-intensive STEM fields and retaining them once there (Belser et al., 2017; Lent et al., 2008). Women and historically underrepresented minorities (URMs) in these fields are especially likely to not take computer science (CS) classes or apply to CS majors or to not persist once enrolled (Sax, Lehman, et al., 2017; Sax, Zimmerman, et al., 2017). Although elementary students are years away from having to declare a major or seek a job, this population is a critical point for learning foundational CS concepts and, perhaps more importantly, how CS practices can be a powerful way of approaching a learning task. Moreover, since positive affective orientation is critical to students maximizing the benefits of these activities, we need to be able gauge their interests, both proximal and distal (Yoo et al., 2017).

Early exposure to high quality computing experiences may inform a young person’s trajectory toward a STEM career, although there are myriad social forces at play that adversely affect girls and students of color having a positive orientation toward either CS or STEM. In fact, children as young as six readily express gendered stereotypes such that boys are better at programming and robotics than girls (Master et al., 2017). Internalization of these beliefs is particularly harmful to girls, as it affects their interest in and self-efficacy for these subjects. Many URMs, historically marginalized in this field, feel unwelcome in or disconnected from CS;

a lower sense of belonging (Johnson, 2011; Leath & Chavous, 2018) and racial/ethnic stereotypes (Margolis et al., 2017) have been cited as reasons why.

A lack of validated attitudinal instruments at the elementary level hampers our ability to both study and address the issue of the gender and racial/ethnic gap in CS. Minimal validation research has been completed on students' CS attitudes that both adheres to core psychological theory and utilizes powerful psychometric analytic methods. Two major exceptions follow. Mason and Rich (2020) represents one of the few serious efforts in this area. They validated their Elementary Student Coding Attitudes Survey (ESCAS)—centered around concepts of coding confidence, interest and utility, in addition to social influence, and perception of coders—using confirmatory factor analysis (CFA) and structural equation modeling (SEM). Despite these efforts, they did not test if their instrument was psychometrically free from gender or race bias. Rachmatullah et al. (2020b) validated a middle grades CS attitudes instrument centered around the concepts of self-efficacy and outcome expectancy by using the combination of classical test theory and item response theory Rasch techniques. This middle grades instrument was analyzed and found psychometrically free of gender and race bias, thus it is a robust starting point we use here for a new instrument to measure and investigate gender and race attitudes at the elementary level.

Theoretical Framework

Bandura (1997) maintained that individuals are motivated by their beliefs in their capabilities to complete a task—called self-efficacy—and that completing that task will ultimately produce a desired outcome, called outcome expectancy. Pajares (1996) argued for task specificity in designing instruments and assessing students' self-efficacy; in our case, the specific task is coding within the domain of computer science. Further, we make use of expectancy-value theory (Eccles & Wigfield, 2002; Wigfield & Eccles, 2000) and its contention that students make purposeful academic decisions based on their expectations for success. In turn, these outcome expectancies influence a student's willingness to select and engage in a task, as well as to persist during challenge.

Related Work

Self-Efficacy and Outcome Expectancy

Prior research on self-efficacy and outcome expectancy beliefs is varied in terms of gender differences. Older literature (e.g. Zeldin et al., 2008) has reported gendered differences in the sources of self-efficacy, with mastery experiences being the primary source of males' self-efficacy and females relying more upon relational information (ie., social input from peers, teachers, parents, and larger society) to inform their self-efficacy. Early practice and success continues to be a persistent factor. In fact, Lishinksi et al. (2016) found that self-efficacy predicted students' course outcomes (ie., exam scores), but gender powerfully affected how students' self-efficacy changed in response to performance feedback early in the course; specifically, early failures or perceived setbacks may prompt female students to disengage from the CS course. It is therefore necessary for educators and researchers to explore this decrease in students' competence beliefs as it greatly affects students' academic performance. Meunks et al's (2018) review of expectancy and competence beliefs indicated that children's expectancy-related beliefs tend to decline from elementary through high school, although students follow different trajectories across different subject areas and these beliefs change based on performance. For elementary-aged female students, self-efficacy is significantly related to their CS career orientation (Aivaloglou & Hermans, 2019).

Research indicates that when students are exposed to negative gender-based stereotypes and they readily endorse those beliefs, their grades and career intentions are affected (Plante et al., 2013). Master and Meltzoff's (2020) extensive review of the literature around stereotypes, STEM, gender, and motivation resulted in their development of a model that underscores the role stereotypes and students' beliefs, attitudes, and behaviors have on their interest and performance in STEM.

Google Inc. and Gallup Inc.'s (2016) report on diversity gaps in CS foregrounds Black/African-American students' higher confidence level and interest in CS compared to White and Hispanic students; this clearly supports the idea that confidence and interest are not the only factors that contribute to (under)representation in the field. DiSalvo et al. (2011) reported that although Black/African-American males enjoyed playing video games, they often did not extend that interest to the computing concepts used to build the games. Findings such as these point to the need for instrument development and further research into the relationship of key demographic factors, beliefs, and outcomes regarding CS education.

Gender and Race/Ethnicity in CS.

There is ample evidence to suggest that CS suffers from a lack of inclusivity. Women and girls often feel unwelcome (Beyer, 2014), suffer from lower confidence (Beyer et al., 2003), or are downright excluded from CS courses and computing in general (Cheryan et al., 2009, 2015). In a landmark study, Sax, Lehman, et al. (2017) found several notable contributors to the gender gap in CS. In particular, they found that women self-reported lower math ability than male counterparts, held a social activist orientation, and felt less compelled to contribute to the scientific community. At the university level, some have found that when CS is taught using a pair programming approach, women perform better and persist in CS courses (Werner et al.,

2004) and self-report higher confidence than those required to work individually (McDowell et al., 2006). Research at the middle school level (Buffum at al., 2015) and elementary level (Tsan et al., 2016) indicates that gender differences are present in students' CS experiences, but how these manifest are quite different. Buffum et al. (2015) found that repeated exposure to CS concepts compensates for differences in students' prior computing experiences, whereas Tsan et al. (2016) found that girls' final CS products were significantly lower in quality compared to all boy groups and mixed gender groups.

Parallelling findings on gender, some research suggests that CS is not particularly open to a range of ethnicities and races. Underrepresented minorities (URMs) often encounter stereotypes about who is 'good' at CS (Margolis et al., 2017), and these stereotypical attributes tend to include high intelligence, limited social skills, and being white or Asian. Moreover, students tend to report that access and wealth positively affects one's ability to participate in CS and that wealth and access are often related to race and ethnicity. As a result, URM often are prevented from developing a sense of belonging in CS which then impedes their interest in pursuing additional coursework, a major, or a career in CS (Sax et al., 2017b). Of particular interest is how the intersection of race and gender might influence a student's CS trajectory; recent findings by Scott and colleagues (2017) highlight how female students of color had lower levels of engagement and interest, stating "...being a member of a marginalized gender group plays a unique role and has a multiplying (negative) effect" (Scott et al., 2017, p. 255). Also of note is that most of this literature focuses on older populations, yet we know (e.g., Aladé et al., 2020; Mulvey & Irwin, 2018) that these social forces start affecting children at younger ages.

Other CS Attitudes Instruments.

Our work has been informed by some notable prior research on CS attitudes. Kukul, Gökçearslan, and Günbatar's (2017) worked with 12 to 14-year-old students in Turkey to produce the Computer Programming Self-Efficacy Scale (CPSES). This 31-item, unidimensional scale queries students on their self-efficacy for specific computing actions, such as "I know where to write the program codes." Self-efficacy, interest, and collaboration drove Kong et al. (2018) to develop and validate a programming empowerment instrument for 4th through 6th grade students. Their 24-item instrument includes statements like "Programming is important to me" and "I like to program with others." More recently, and as noted earlier, Mason and Rich (2020) validated their Elementary Student Coding Attitudes Survey. This 23-item, 5-factor instrument queries 4th through 6th grade students on statements such as "Coding is interesting" and "Kids who code are smarter than average." All three of these instruments fall short of what is needed for evaluating young students' CS attitudes, despite analyzing children's responses around the same grade bands. They are all quite lengthy at 23 to 31 items and none of them evaluated if the instrument was free from bias. Moreover, the Kukul et al., (2017) instrument has not been validated in English.

Current Work

There is a clear need to research students' attitudes by race/ethnicity and gender, especially beginning at a young age. Per the review above, there is a lack of a brief, targeted, validated, and psychometrically bias-free instrument that measures CS Attitudes in upper elementary students and which accounts for the unique developmental differences of this population. To account for this, we detail below our qualitative procedures for ensuring young students understood our item wording (Vandenberg et al., 2020), after which we follow similar validation procedures as Rachmatullah et al. (2020b). This instrument, the E-CSA, is then used to measure upper elementary (4th and 5th grade) students' attitudes toward computer science, with particular focus on the effect of race/ethnicity and gender on their responses. The following research questions guide this investigation:

1. With regards to the validation of the instrument, what model best represents the dimensionality and internal structure of E-CSA?
2. What is the relationship between elementary students' CS attitudes, measured on the E-CSA, and their CS performance, measured on the E-CSCA (Elementary Computer Science Concepts Assessment)?
3. What is the influence of race/ethnicity and gender on students' responses on the E-CSA instrument?

Methodology

Item Development

The items for our instrument were based on the previously validated Engineering and Technology attitudes subscale of the Student Attitudes toward STEM (S-STEM) Survey (Friday Institute, 2012). The S-STEM survey has been used with over 15,000 4th through 12th grade US students (Wiebe et al., 2018). We then engaged in an iterative process of cognitive interviews with a diverse group of 98 4th and 5th grade students on their understanding of the items (Vandenberg et al., 2020). Findings from this process indicated that upper elementary aged students conceptualized of *doing* computer science as 'coding.' To make this lean instrument appropriate for young students, we privileged their words and the types of tasks in which they engaged in what we, as researchers and practitioners, consider computer science. As such, we used the word `coding' because children were not able to define computer science. This resulted in a final set of 11 Likert-scale items with 5 points from *strongly disagree* to *strongly agree* that reflected the modified wording of coding and computer science rather than engineering and technology. This instrument, the E-CSA, is based on two psychological constructs, self-efficacy (denoted as SE_) and outcome expectancy (denoted as OE_) (see Table 1).

Table 1

Elementary CS Attitudes (E-CSA) Instrument Items

Item Number	Item Wording	Construct
SE_1	I would like to use coding to make something new.	Self-efficacy
OE_1	If I learn coding, then I can improve things that people use everyday.	Outcome expectancy
SE_2	I am good at building code.	Self-efficacy
SE_3	I am good at fixing code.	Self-efficacy
OE_2	I am interested in what makes computer programs work.	Outcome expectancy
OE_3	Using code will be important in my future jobs.	Outcome expectancy
OE_4	I want to use coding to be more creative in my future jobs.	Outcome expectancy
OE_5	Knowing how to code computer programs will help me in math.	Outcome expectancy
OE_6	Knowing how to code computer programs will help me in engineering.	Outcome expectancy
OE_7	Knowing how to code computer programs will help me in science.	Outcome expectancy
SE_4	I believe I can be successful in coding.	Self-efficacy

Sample and Contexts

Following university IRB approval, which required both parental consent and minor student assent, a total of 169 students consented/assented to take the E-CSA instrument as part of either a classroom-based study or a standalone survey administration for the purposes of this validation. This number is sufficient to perform IRT Rasch and obtain stable item calibrations and person measure estimates (Chen et al., 2014; Linacre, 1994). Participating students were third through fifth grade students (ages 8-11), with 5th grade students representing 66% of the sample and female students accounting for approximately 54% of the sample. White/Caucasian students were the most commonly reported ethnicity/race, with almost 59%. For analysis here, and in alignment with the demographic profile of the CS community, White and Asian students comprise our non-URM category, with Black/African-American, Hispanic/Latino, Native American/American Indian, multiracial, and other comprising URM (Beede et al., 2011; National Academies of Sciences, Engineering, and Medicine, 2018; Smith et al., 2018; Wiebe et al., 2018). Full sample demographics are reported in Table 2.

Standalone survey administration began in summer 2020 and included a virtual summer camp and remote classroom administration due to COVID-19. The virtual summer camp emphasized engineering topics for students in grades 3 to 5. Our survey served as a consent-only final activity the campers completed after a weeklong camp session. Remote survey administration through classroom teachers began in August 2020; none of the participating teachers were technology specialists, but rather 4th or 5th grade teachers who provided the parents with consent documents and followed up with consented and assented students. All of these students took the E-CSA instrument one time.

The classroom-based studies occurred in fall 2019 and February and March of 2020 (pre-pandemic) and participating students were expected to complete the E-CSA both before and after the intervention. The fall 2019 study involved block-based coding instruction across three pair programming conditions, assigned at the classroom level. The three conditions were traditional pair programming with one computer and related driver-navigator roles, two computers without roles, and two computers with roles (Vandenberg et al., 2021). This study included only 5th grade students and lasted four weeks. The spring 2020 study took place over five weeks and involved implementing and comparing four system-based features to encourage 4th and 5th grade students using traditional pair programming to transfer the driver-navigator roles appropriately and to talk to their partner more effectively.

Table 2

Table of Participant Self-Reported Demographics

Individual-level Variables	N	Percentage (%)
Age		
8	7	5
9	36	23
10	85	54
11	28	18
Gender		
Male	73	43
Female	92	54
No Response	4	3
Grade		
3rd	5	3
4th	52	31

Table 2 (continued)

5th	112	66
Race/Ethnicity		
White	100	59
Black/ African American	14	9
Hispanic/Latino	15	9
Asian	12	7
Native American/ American Indian	3	2
Other	2	1
Multiracial	12	7
No Response	10	6
Data Collected		
Classroom-based (pre and post)	70	41
Standalone (one time only)	99	59

Validation Procedure

To answer Research Question 1, we conducted a validation of the developed instrument. The validation procedure used in this study was based on the Standards for Educational and Psychological Testing proposed by the American Educational Research Association, American Psychological Association, and National Council on Measurement in Education (AERA, APA, & NCME, 2014). This standard suggests five significant points on which to validate an instrument: response processes, test content, internal structure, consequences of testing, criterion validity, or the relationship between the measured constructs and other theoretically related variables. Given that previous work addressed the first two points (Vandenberg et al., 2020), the current study focused on the three latter points.

A combination of the classical test theory and item response theory-Rasch approaches was used in this study to examine the psychometric model and internal structure of the E-CSA (cf., Rachmatullah et al., 2020a, 2020b).We started by evaluating the number of latent constructs (dimensionality) and the items' quality within the instrument. Two models were compared in this dimensionality analysis: one- and two-dimensional (factor, construct, and dimension used interchangeably throughout this paper) models. A one-dimensional model which groups all items in a single factor was used as the baseline. The two-dimensional model was based on our theoretical conceptualization of the instrument based on two factors: self-efficacy and outcome expectancy (Bandura, 1986; Eccles & Wigfield, 2002; Wigfield & Eccles, 2000). Adams and Wu's (2010) procedure was used to identify the best-fitting model which is the model that has the lowest values in final deviance across three criteria: Akaike Information Criterion (AIC), Akaike Information Criterion Corrected (AICc), and Bayesian Information Criterion (BIC). Mean-square (MNSQ) values were used to assess the item quality, with the assumption that a well-behaved high quality item has an MNSQ value ranging from 0.60 to 1.40 (Wright and Linacre, 1994). All of these analyses were run in ConQuest version 5.12.3 (Adams et al., 2020).

Differential Item Functioning (DIF) was run on the instrument as part of our IRT methods to examine item bias. DIF analysis is used to evaluate whether a certain group member in the study has different probabilities of endorsing a certain item controlling for the overall score (Boone & Scantlebury, 2006; Boone et al., 2014). An item exhibiting DIF indicates a bias towards a particular group (Boone & Scantlebury, 2006; Boone et al., 2014), such as with gender, males tend to agree more on items about sports than females. In other words, an individual item that is biased does not automatically warrant removal of the item, but an instrument that has DIF items may lead to interpretation issues with regards to the problematic demographic factor (Boone & Scantlebury, 2006; Boone et al., 2014). DIF analysis addresses what Messick (1995) called the generalizability aspect of construct validity. Gender (Male and Female) and majority group representation (URM vs. non-URM) are of considerable interest to the computer science education research and policy community (e.g., Belser et al., 2017; Beyer, 2014; Sax, Lehman, et al., 2017; Sax, Zimmerman, et al., 2017) and thus will be the focus of our DIF analysis. We used the cut-off value of < 0.64, as suggested by Boone et al. (2014), to evaluate the DIF of an item. An item that had a DIF contrast value more than the cutoff, it would be demonstrating unacceptable bias based on the demographic factor of interest and is typically removed.

After all the problematic items were removed, a CFA was then run to provide additional structural validity evidence. Informed by the results of the initial dimensionality analysis, we only ran CFA on the two-factor model and evaluated this model using the cut-off suggested by Hair et al. (2019). The acceptable model should have chi-square/df < 3, root mean square error of approximation (RMSEA) < .08, comparative fit index (CFI) > .95, and Tucker–Lewis index (TLI) > .95. CFA was performed in IBM SPSS Amos version 26 (Arbuckle, 2019).

Lastly, ensuring the internal consistency and accuracy of the item responses was done by evaluating the reliability values. Three different reliability values were computed using the CTT and IRT-Rasch methods, namely Cronbach's alpha, item separation reliability, and person reliability (person/a posteriori plausible value reliability). Linacre (2012) suggested that the two latter reliabilities can be evaluated in the same way as Cronbach's alpha. Thus we used the same cut-off value of > .70 to determine an acceptable reliability value (DeVellis, 2017).

Data Analysis

Students' raw scores were converted using Rasch analysis to obtain scores in the ratio-interval form (logit). Each student had scores for both of the constructs, self-efficacy and outcome expectancy. We used these logit scores for the subsequent analyses. A Pearson correlation test was run to examine the relationship between CS attitudes—self-efficacy and outcome expectancy—and conceptual understanding of CS. An instrument, the E-CSCA (Elementary Computer Science Concepts Assessment; Vandenberg et al., 2021) adopted from Rachmatullah et al. (2020a) was used to measure students' conceptual understanding of CS. Two example items from the E-CSCA appear in the Appendix; these were based on the work of Rachmatullah et al (2020a) and recently validated in Vandenberg et al. (2021). Furthermore, a two-way ANCOVA test was run to explore the interaction effect of gender and race/ethnicity on elementary students' CS self-efficacy and outcome expectancy, by controlling for test occasions (pre-posttest or standalone). Tests of simple slopes were run to decompose the interaction effect. The effect sizes were calculated using Cohen's *d*, with 0.20, 0.50 and 0.80 for small, medium and large effect sizes respectively (Lakens, 2013). All these analyses were performed using the **lm** package in RStudio (RStudio Team, 2020).

Results

Instrument Validation

Multidimensional Rasch Analysis and Item Fit-Statistics

Multidimensional Rasch analysis was run to assess the best fitting model of E-CSA. Table 3 presents the results of the multidimensional Rasch analysis. We found that both one- and two-dimensional models of E-CSA did not have any misfitting items. However, the two-dimensional model had lower (ie., better) values of the final deviance criteria (AIC, AICc, and

BIC) than the one-dimensional model. A Chi-square test on the AIC showed a significant difference between one- and two-dimensional models (X2 = 56.95, $p < .05$), indicating that the two-dimensional model was the best model. We then used this two-dimensional model in our subsequent analysis.

Table 3

Comparison between One- and Two-dimensional Models of E-CSA

Model	X2	df	Final Deviance	AIC	AICc	BIC	Number of Parameter	Number of Misfitting Items
One-dimension	199.51	10	6521.66	6551.66	6549.95	6603.30	15	0
Two-dimension	209.08	9	6460.71	6494.71	6492.51	6553.23	17	0

Table 4 shows the fit statistics for all items in the two-dimensional model representing both constructs–CS self-efficacy and CS outcome expectancy. All the items had weighted and unweighted MNSQ values within the range of acceptable values, 0.60 -1.40, as suggested by Wright and Linacre (1994). These values demonstrated that the items were psychometrically sound and able to differentiate students based on the degree of their CS self-efficacy and outcome expectancy. Moreover, a Wright map (see Appendix) from the multidimensional analysis shows a reasonable distribution of students' CS self-efficacy and outcome expectancy responses, from strongly disagree (below Level 1) to strongly agree (above Level 4).

Table 4

Item Fit Statistics for The Two-Dimensional E-CSA Model

Construct	Item Code	Estimate	Weighted MNSQ	Unweighted MNSQ	DIF Gender	DIF Race	Alpha if Item Deleted
CS Self-Efficacy	SE_1	-0.369	1.02	0.94	0.21	0.09	0.794
	SE_2	0.306	0.89	0.85	0.09	0.02	0.716
	SE_3	0.541	1.05	1.05	0.06	0.09	0.754
	SE_4	-0.479	0.96	0.89	0.02	0.06	0.785
CS Outcome Expectancy	OE_1	-0.253	1.05	1.16	0.19	0.06	0.818
	OE_2	0.033	1.20	1.24	0.63	0.04	0.814
	OE_3	0.294	1.07	1.06	0.46	0.21	0.822
	OE_4	0.372	0.82	0.81	0.18	0.20	0.793
	OE_5	0.158	0.91	0.95	0.26	0.02	0.811
	OE_6	-0.490	1.23	1.18	0.46	***0.83****	0.826
	OE_7	-0.114	1.09	1.13	0.49	0.17	0.828

Note: * See sections “Differential Item Functioning - Gender and Race” and “Discussion, Research Question 1” for more about how to interpret results using this item.

Differential Item Functioning - Gender and Race/Ethnicity

We also ran DIF analyses for gender and race to address the generalizability aspect of construct validity. The results indicated that most of the items were free from gender and race-bias, suggesting that they behaved equally to all gender and race groups. We only detected one item with a DIF for race/ethnicity (URM/non-URM), OE_6, with a DIF contrast value of 0.83. We chose not to remove this item, as other psychometric indices indicated it was a good quality item. However, we suggest carefully interpreting the results using this item when conducting analyses comparing CS outcome expectancy by race/ethnicity. Table 4 presents the results of DIF gender and race/ethnicity analyses.

CFA

The structural model of the E-CS Attitude was then analyzed through CFA. All the original items were included in the CFA, as multidimensional Rasch and DIF analyses indicated no problematic items. We compared two models: the model without correlated residual errors (Model 1) and correlated residual errors (Model 2). For Model 2, the correlated residuals were determined based on the modification indices and the items' context (Hair et al., 2019). After evaluating all the fit statistics indicators, we found that Model 2 had significantly better fit statistics than Model 1. Table 5 shows all the fit statistics for these two models with Model 2 demonstrating lower chi-square and RMSEA and higher CFI and TLI, and therefore better values (see cut off values in Table 5 next to the indicators), and Figure 1 visualizes the structure of the E-CSA two-factor model with correlated residual errors.

Table 5

Comparing CFA Models with and without Correlated Residuals (target parameter values in parentheses)

Indicator	Model 1	Model 2
X2	174.58	73.68

Table 5 (continued)

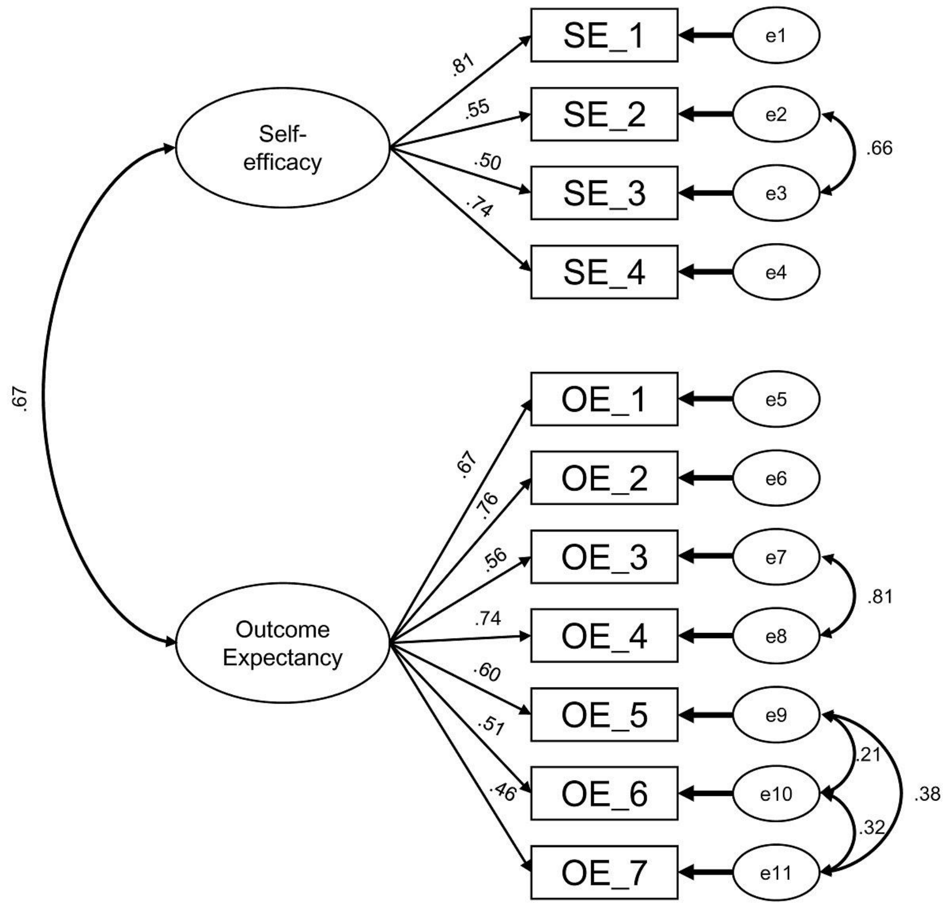
df	42	38
X2/df (<3)	4.16	1.94
p-value	< .001	< .001
CFI (> .95)	.884	.969
TLI (>.95)	.818	.946
RMSEA (< .08)	.117	.064
ΔX2(Δdf)	-	100.90
p-value forΔX2	-	< .001

Reliability

Cronbach's alpha and plausible value (PV; aka person reliability) generated from the multidimensional Rasch analysis were used to assess the internal consistency of the E-CS Attitudes. The CS self-efficacy construct had Cronbach's alpha and PV reliability values of .812 and .843, respectively. For the CS outcome expectancy, the Cronbach's alpha and PV reliability values were .838 and .883, respectively. All of these values were above the acceptable value of .70 (DeVellis, 2017), indicating a stable instrument. Also, a separation reliability value was computed through multidimensional Rasch analysis, evaluating the reproducibility of the spread of the response levels. The separation reliability for the E-CSA was .960, indicating a good spread of item response.

Figure 1.

Final CFA Model (Model 2) for Two-Factor E-CSA with Standardized Loadings



Note: The figure above demonstrates that self-efficacy and outcome expectancy are two distinct factors (see section "Multidimensional Rasch Analysis and Item Fit-Statistics"), the individual items associated with the factors (see Table 1), and the correlated residuals indicated by the double headed arrows on the right (see sections "CFA" and "Discussion, Research Question 1")

Correlation between CS Attitude and CS Performance

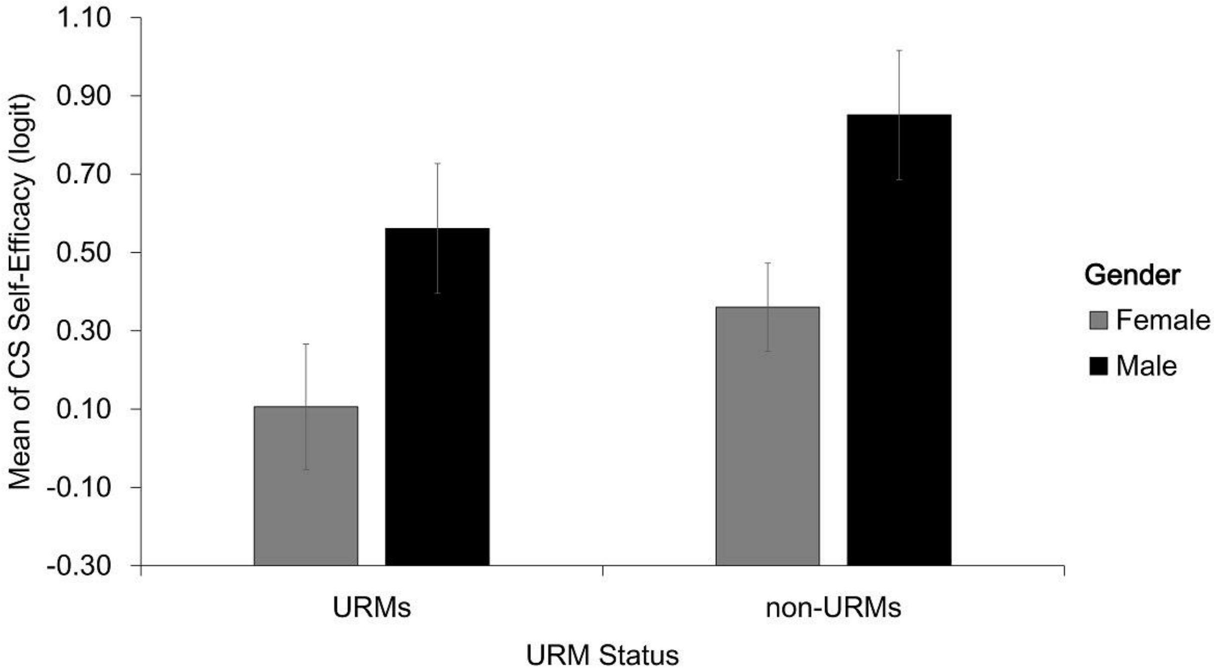
To address Research Question 2, Pearson correlation tests were run to examine the correlation between the two constructs in the E-CS Attitudes and students' conceptual understanding of CS. We found that CS self-efficacy had a significant positive correlation (r = .24, p = .002) with the CS conceptual understanding. In contrast, we did not find a significant correlation between CS outcome expectancy and CS conceptual understanding (*r* = .09, *p* = .278).

Interaction Effect between Gender and Race/Ethnicity on Elementary CS Attitude

To address Research Question 3, two-way ANCOVA tests were performed to examine the interaction effect of gender and race/ethnicity (based on URM vs. non-URM) on elementary students' CS self-efficacy and outcome expectancy. For CS self-efficacy, we found that the interaction effect between gender and race/ethnicity was not significant after controlling for test occasion (pre-posttest or standalone; *t* = 0.03, *p* = .920). We then removed the interaction effect from the model, and ran another model in which we found that gender had a significant fixed effect on elementary students' CS self-efficacy with a small effect size (*t* = 3.15, *p* = .002, *d* = .11). Decomposing this result, male students (M = 0.79, SD = 0.69) had higher CS self-efficacy than female students (*M* = 0.32, *SD* = 0.93). In contrast, there was a non-significant fixed effect of race/ethnicity on CS self-efficacy (*t* = 0.27, *p* = .11, *d* = 0.22) indicating non-URM students (*M* = 0.55, *SD* = 1.21) did not differ from URM students (*M* = 0.36, *SD* = 0.90). The results are visualized in Figure 2.

Figure 2.

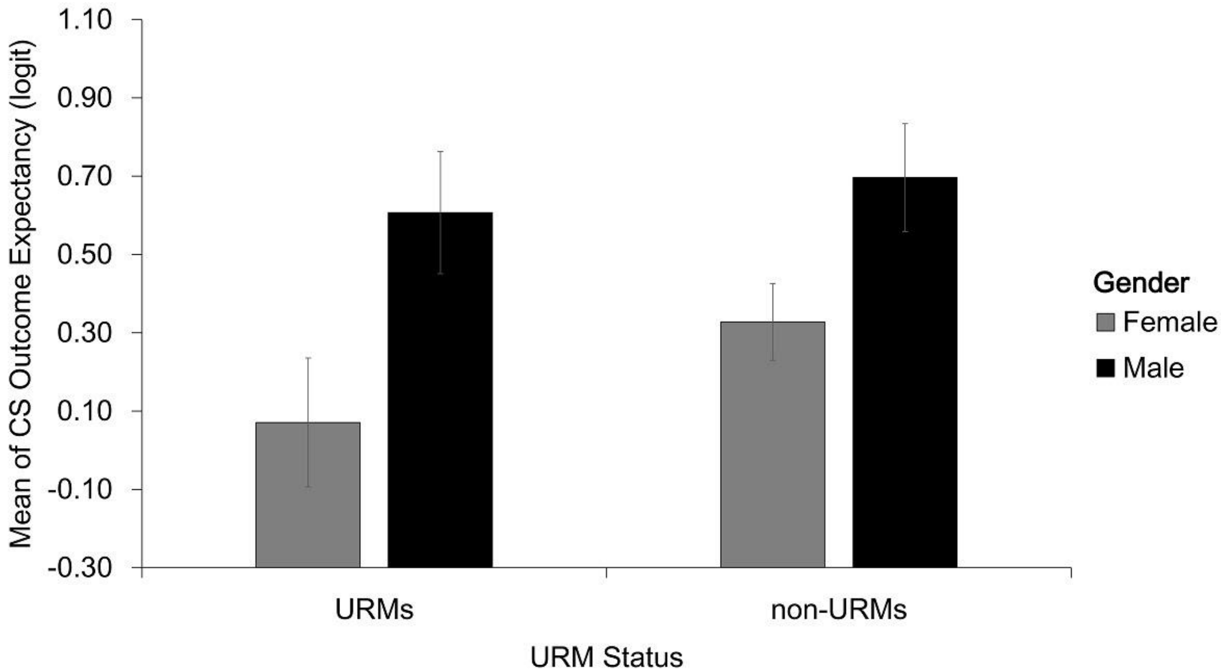
Differences in CS Self-efficacy based on Gender and Race/Ethnicity



Similar to the findings in CS self-efficacy, the interaction effect of gender and race/ethnicity was not significant on the CS outcome expectancy (t = 0.17, p = .577). After we removed this interaction effect from the model, we again found a significant fixed effect of gender on CS outcome expectancy (t = 3.05, p = .002, d = .04), where male students (M = 0.69, SD = 1.02) had higher scores than female students (M = 0.27, SD = 0.93). As with the findings for CS self-efficacy, we also did not find a significant fixed effect of race/ethnicity on CS outcome expectancy (t = 0.16, p = .25, d = .33). This indicated that non-URM students (M = 0.47, SD = 1.02) did not differ from URM students (M = 0.37, SD = 0.92) in CS outcome expectancy. Figure 3 presents the results.

Figure 3

Differences in CS Outcome Expectancy based on Gender and Race



Discussion

In order to address gaps in CS participation, from elementary classrooms to university major enrollment, it is important to explore students' attitudes (self-efficacy and outcome expectancy) towards CS, and to what extent differences in attitude appear by race/ethnicity and gender. As such, we set out to examine these differences through a brief bias-free instrument we developed and validated, and appropriate for upper elementary student use. We discuss our findings by research question.

Research Question 1: With regard to the validation of the instrument, what model best represents the dimensionality and internal structure of the E-CS Attitudes?

In this study, we achieved content validity by relying on prior scholarly work on self-efficacy (Bandura, 1997) and outcome expectancy (Eccles & Wigfield, 2002; Wigfield & Eccles,

2000) and by utilizing and modifying a previously validated instrument (S-STEM; Unfried, et al., 2015). Moreover, prior work of ours made use of the rigorous process of cognitively interviewing a diverse array of upper elementary students on their understanding of the terminology used in the instrument items (Vandenberg et al., 2020). Others (Padilla & Benitez, 2014; Wilson & Miller, 2014) have utilized this approach to ensure content validity based on response processes.

Regarding consistencies in test responses, as typically evaluated through reliability values, we utilized Cronbach's alpha and PV reliability values generated through CTT and IRT methods. Our results indicate a stable instrument in which participants consistently responded to the items within each factor in a relatively similar fashion.

Additionally, we explored the instrument and individual item quality through confirmatory factor analysis (CFA) and multidimensional Rasch modeling. Having established an *a priori* hypothesis about the latent factors and variables, owing in large part to basing our work on a previously validated and theoretically-sound instrument, we were confident in beginning our classical test theory work with a CFA (e.g., Thompson, 2004). Our CFA and multidimensional Rasch modeling resulted in a two-factor model as best fit, aligning with our *a priori* expectations based on the theoretical framework upon which the instrument is based. These assumptions were supported, as the two-factor model with correlated residuals had significantly better fit statistics. This type of post hoc model fitting is used when theoretically and meaningfully justified (Brown, 2003, 2015), such as when covariance occurs due to content overlap or item phrasing. In our case, we identified sets of items whose content/wording and sequential proximity may have influenced how students responded to them. In particular, SE_2 and SE_3, "I am good at building code" and "I am good at fixing code", respectively, appear in sequence and have extremely similar wording. Additionally, OE_3 and OE_4, “Using code will be important in my future jobs” and “I want to use coding to be more creative in my future jobs”, respectively, also appear in sequence and both reference “future jobs”. Lastly, we permitted the residuals of the following to correlate: “Knowing how to code computer programs will help me in ____” math (OE_5), engineering (OE_6), and science (OE_7). By allowing these terms to correlate, our fit indices improved to acceptable levels.

Lastly, regarding generalizability, we completed DIF analysis to explore the fairness of the instrument across varied socio-demographic subgroups of students. Our results indicate that the instrument is largely free, psychometrically, from gender and race bias, with one item (OE_6) demonstrating marginal DIF by race/ethnicity. This was near the threshold for removal (Boone et al., 2014); we opted to retain the item, but users of the instrument need to be aware. This item, “Knowing how to code computer programs will help me in engineering,” was one where we found marked qualitative differences in students’ responses during the cognitive interview process. Based on our prior work (Vandenberg et al., 2020), students in rural, underserved, low SES, and largely Black/African-American and Latinx schools struggled to provide a robust definition for engineering. Therefore, the scores computed from E-CSA’s outcome expectancy scale should be carefully interpreted when comparing groups based on race/ethnicity on this scale. We believe that this problematic item highlights the need for more substantive exposure to engineering education and experiences at the elementary level for all children.

Research Question 2: What is the relationship between elementary students' CS attitudes and their CS performance?

In this study, we treated the students' scores on a measure of conceptual CS understanding as being theoretically related to CS self-efficacy and outcome expectancy measured via E-CSA. Our results indicate that of the two factors that comprise the E-CSA, only self-efficacy was significantly positively correlated with conceptual understanding. Self-efficacy has long been considered a predictor of student outcomes (eg., Bandura, 1986; Brosnan, 1998; Lishinski et al., 2016); these empirical findings align with our own. Research indicates that there is positive impact of (CS-related) experience on self-efficacy (Bandura, 1986; Hinckle et al., 2020). Further, by improving CS self-efficacy, we can expect to see improvements in CS performance. Although outcome expectancy was not correlated with student scores on the E-CSA, we believe it is still a valuable measure, as it compliments self-efficacy in providing a more complete motivational model of the student with regards to CS (Eccles & Wigfield, 2002).

Research Question 3: What is the influence of race/ethnicity and gender on students' responses on the E-CSA instrument?

We found that gender had a statistically significant effect on CS Attitudes, with males having higher self-efficacy and outcome expectancy than females. This difference has profound implications for both proximal and distal interests and performance and it is not a new issue. Empirical research from the 1990s indicated that males self-report higher confidence for, more liking of, and lower anxiety with computers (Charlton, 1999; Colley et al., 1994). Newer research largely mirrors earlier findings (Beyer, 2014; Wilcox & Lionelle, 2018), although there may be reason to hope as these findings likely indicate a path forward for girls. In particular, Schmidt (2011) found that females' lower interest in technology leads to reduced experience and knowledge. Contrast that with Wilcox and Lionelle (2018) who note that female students outperform their male peers when they have similar levels of prior computing experience. Providing girls and young women with consistent and quality computing experiences is essential. This needs to be addressed through concerted efforts at even younger grades than we tested here in order to try to prevent the development of these deleterious gender-based attitudes. In addition, it is noteworthy that most previous studies investigating gender differences in elementary or upper education level students' attitudes towards CS did not examine whether the instrument used in those studies were free from bias (cf., Kong et al., 2018; Mason & Rich, 2020). Thus, the results may not be valid with regards to research questions centering on gender. We believe our findings on gender differences in elementary CS attitudes towards CS have rigorously addressed this potential problem with item bias, as DIF analysis indicated that our instrument items were free of gender bias.

It is also important to note that there was a nonsignificant effect of race/ethnicity on CS self-efficacy and outcome expectancy. Based on our findings, URM and non-URM students did not differ in these constructs. This is meaningful as other research indicates that URMs often indicate lower interest in CS and generally find CS to be an unwelcoming place (Margolis et al., 2017; Scott et al., 2017). That we did not find a statistically significant difference is intriguing. It could be that these young students have not yet encountered negative racial and ethnic stereotypes that might influence their perceptions of themselves. Most prior research on racial and ethnic stereotypes have occurred with older students (e.g., Johnson, 2011; Margolis et al., 2017; Scott et al., 2017); however, a recent study found that young children, ages 3 to 8, did not use racial/ethnic information to make decisions about who ought to perform certain STEM-based jobs (Mulvey & Irvin, 2018).

Limitations

We acknowledge the following limitations of this study and suggest them for future work as part of both the instrument development process and examining gender and race/ethnicity in CS education. First, we had a relatively small sample size (N=169) of students who completed the E-CS instrument. To compensate for this, we used robust, psychometrically sound techniques for this smaller sample size. However, the students who did participate were largely white (59.2%) and thus limited our ability to explore the relationship of race/ethnicity to both attitudinal and learning factors. Future work would benefit from a more diverse racial and ethnic sample. It is worth noting that grouping by URM and non-URM is not the only approach that can be used to examine effects of race and ethnicity. While it increases sub-sample size (and related statistical power) to group multiple demographic categories, it can also mask important patterns happening at a finer-grained level. Relatedly, despite purposeful sampling across diverse school populations and contexts, all results are from a single state in the United States. Future work would benefit from more widespread national and international data collection and with populations with various levels of CS-related experiences. Additionally, we did not account for teacher- or school-level differences; future work with a more substantive sample size might consider conducting multilevel modeling to explore this further (Lee, 2000). Finally, survey item order was set, perhaps contributing to the nonrandom errors in the model. Future administrations should consider randomizing the items to test for and reduce this effect.

Conclusion

This study examined gender and race differences in elementary students' attitudes towards CS. To that end, we developed and validated a survey called Elementary Computer Science Attitudes (E-CSA) which consisted of the constructs of CS self-efficacy and outcome expectancy, through a combination of classical test theory (CTT) and item response theory (IRT) Rasch. The E-CS was found to be, psychometrically, a gender and race bias-free instrument. We found no significant interaction effect between gender and race in the two constructs of CS Attitudes. We also did not see a significant difference based on race. However, a significant difference was found in both CS attitudes constructs based on gender, whereby male students had higher CS attitudes than female students.

Prior work has established the link between students' beliefs, such as their self-efficacy for a content area, and their performance in that area (Brosnan, 1998; Lishinski et al., 2016). Having an instrument that assesses students' attitudes toward computer science that is based on theoretically-derived constructs, self-efficacy and outcome expectancy, could prove indispensable to researchers and practitioners alike. To this end, we developed and rigorously validated a brief instrument appropriate for use with upper elementary students.

We believe that use of the instrument can inform classroom-based interventions, the development of curricular materials, and reinforce findings from other cross-sectional CS studies. In particular, we believe that our findings support the need for early and consistent CS interventions with girls (see Happe et al., 2020; Hur et al., 2017) so as to support their positive attitudes toward CS. Moreover, as the instrument was validated with upper elementary students, we support the use of it alongside other analyses with the same aged population.

In addition to addressing the limitations noted above, future research could explore how CS attitudes correlate with non-STEM subject areas. Prior work of ours indicated that some students understood the CS concept of debugging to be much like editing and revising a paper in a writing class. Additionally, we are interested in how remote learning and the increased use of technology may have implications for students' interest in CS. And lastly, future work could explore how students' talk about CS and coding may reflect their beliefs and overall interests.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No DRL 1721160. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Adams, R., & Wu, M. (2010). *Notes and tutorial ConQuest: Multidimensional model*. Retrieved from https://www.acer.org/ conquest/notes-tutorials

Adams, R.J, Wu, M.L, Cloney, D., and Wilson, M.R. (2020). *ACER ConQuest: Generalised Item Response Modelling Software* [Computer software]. Version 5. Camberwell, Victoria: Australian Council for Educational Research.

Aivaloglou, E., & Hermans, F. (2019, February). Early programming education and career orientation: the effects of gender, self-efficacy, motivation and stereotypes. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 679-685).

Aladé, F., Lauricella, A., Kumar, Y., & Wartella, E. (2020). Who's modeling STEM for kids? A character analysis of children's STEM-focused television in the US. *Journal of Children and Media,* 1-20.

American Educational Research Association, American Psychological Association, National Council on Measurement in Education. (2014*). Standards for educational and psychological testing*. Washington, DC: AERA.

Arbuckle, J. L. (2019). *Amos* (Version 26.0) [Computer Program]. Chicago: SPSS.

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Englewood Cliffs, NJ: Prentice Hall.

Bandura, A. (1997). *Self-efficacy: The exercise of control*. New York: Freeman.

Belser, C. T., Prescod, D. J., Daire, A. P., Dagley, M. A., & Young, C. Y. (2017). Predicting undergraduate student retention in STEM majors based on career development factors. *The Career Development Quarterly, 65*(1), 88-93.

Beyer, S. (2014). Why are women underrepresented in Computer Science? Gender differences in stereotypes, self-efficacy, values, and interests and predictors of future CS course-taking and grades. *Computer Science Education*, *24*(2-3), 153-192.

Beyer, S., Rynes, K., Perrault, J., Hay, K., & Haller, S. (2003, January). Gender differences in computer science students. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (pp. 49-53).

Boone, W. J., & Scantlebury, K. (2006). The role of Rasch analysis when conducting science education research utilizing multiple-choice tests. *Science Education*, *90*(2), 253-269.

Boone, W. J., Staver, J. R., & Yale, M. S. (2014). *Rasch analysis in the human sciences*. Dordrecht, Netherlands: Springer.

Brosnan, M. J. (1998). The impact of computer anxiety and self-efficacy upon performance. *Journal of Computer Assisted Learning*, *14*(3), 223-234.

Brown, T. (2003). Confirmatory factor analysis of the Penn State Worry Questionnaire: Multiple factors or method effects? *Behaviour Research and Therapy, 41*, 1411-1426.

Brown, T. (2015). *Confirmatory factor analysis for applied research*. New York, NY: Guilford Press.

Buffum, P. S., Frankosky, M., Boyer, K. E., Wiebe, E., Mott, B., & Lester, J. (2015, August). Leveraging collaboration to improve gender equity in a game-based learning environment for middle school computer science. In *2015 Research in Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)* (pp. 1-8). IEEE.

Charlton, J. P. (1999). Biological sex, sex-role identity, and the spectrum of computing orientations: A re-appraisal at the end of the 90s. *Journal of Educational Computing Research, 21*(4), 393-412.

Chen, W. H., Lenderking, W., Jin, Y., Wyrwich, K. W., Gelhorn, H., & Revicki, D. A. (2014). Is Rasch model analysis applicable in small sample size pilot studies for assessing item characteristics? An example using PROMIS pain behavior item bank data. *Quality of Life Research, 23*(2), 485-493.

Cheryan, S., Master, A., & Meltzoff, A. N. (2015). Cultural stereotypes as gatekeepers: Increasing girls' interest in computer science and engineering by diversifying stereotypes. *Frontiers in Psychology, 6,* 49.

Cheryan, S., Plaut, V. C., Davies, P. G., & Steele, C. M. (2009). Ambient belonging: how stereotypical cues impact gender participation in computer science. *Journal of Personality and Social Psychology, 97*(6), 1045.

Colley, A. M., Gale, M. T., & Harris, T. A. (1994). Effects of gender role identity and experience on computer attitude components. *Journal of Educational Computing Research, 10*(2), 129-137.

DeVellis, R. F. (2017). *Scale development: Theory and applications* (4th ed.). Los Angeles, CA: Sage.

DiSalvo, B., Yardi, S., Guzdial, M., McKlin, T., Meadows, C., Perry, K., & Bruckman, A. (2011, May). African American men constructing computing identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2967-2970).

Eccles, J. S., & Wigfield, A. (2002). Motivational beliefs, values, and goals. *Annual Review of Psychology, 53*(1), 109-132.

Friday Institute for Educational Innovation (2012). *Middle and high school STEM-Student survey*. Raleigh, NC: Author.

Google Inc. & Gallup Inc. (2016). *Diversity gaps in computer science: Exploring the underrepresentation of girls, Blacks and Hispanics*. Retrieved from http://goo.gl/PG34aH.

Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2019). *Multivariate data analysis: Eight edition*. Hampshire, UK: Cengage Learning.

Happe, L., Buhnova, B., Koziolek, A., & Wagner, I. (2020). Effective measures to foster girls' interest in secondary computer science education. *Education and Information Technologies*, 1-19.

Hinckle, M., Rachmatullah, A., Mott, B., Boyer, K. E., Lester, J., & Wiebe, E. (2020, June). The Relationship of Gender, Experiential, and Psychological Factors to Achievement in Computer Science. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 225-231).

Hur, J. W., Andrzejewski, C. E., & Marghitu, D. (2017). Girls and computer science: experiences, perceptions, and career aspirations. *Computer Science Education, 27*(2), 100-120.

Johnson, D. R. (2011). Examining sense of belonging and campus racial diversity experiences among women of color in STEM living-learning programs. *Journal of Women and Minorities in Science and Engineering, 17*(3).

Kong, S. C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education, 127*, 178-189.

Kukul, V., Gökçearslan, Ş., & Günbatar, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Eğitim Teknolojisi Kuram ve Uygulama, 7*(1), 158-179.

Lakens, D. (2013). Calculating and reporting effect sizes to facilitate cumulative science: a practical primer for t-tests and ANOVAs. *Frontiers in Psychology, 4*, 863.

Leath, S., & Chavous, T. (2018). Black women's experiences of campus racial climate and stigma at predominantly White institutions: Insights from a comparative and within-group approach for STEM and non-STEM majors. *The Journal of Negro Education, 87*(2), 125-139.

Lee, V. E. (2000). Using hierarchical linear modeling to study social contexts: The case of school effects. *Educational Psychologist, 35*, 125–141.

Lent, R. W., Lopez Jr, A. M., Lopez, F. G., & Sheu, H. B. (2008). Social cognitive career theory and the prediction of interests and choice goals in the computing disciplines. *Journal of Vocational Behavior, 73*(1), 52-62.

Linacre, J. M. (2012). *A user's guide to Winstep*. Ministep Rasch-Model Computer Programs: Program Manual 3.73. 0. 2011.

Lishinski, A., Yadav, A., Good, J., & Enbody, R. (2016, August). Learning to program: Gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance. In *Proceedings of the 2016 ACM Conference on International Computing Education Research* (pp. 211-220).

Margolis, J., Estrella, R., Goode, J., Holme, J. J., & Nao, K. (2017*). Stuck in the shallow end: Education, race, and computing*. MIT press.

Mason, S. L., & Rich, P. J. (2020). Development and analysis of the Elementary Student Coding Attitudes Survey. *Computers & Education*, 103898.

Master, A., Cheryan, S., Moscatelli, A., & Meltzoff, A. N. (2017). Programming experience promotes higher STEM motivation among first-grade girls. *Journal of Experimental Child Psychology, 160*, 92-106.

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM, 49*(8), 90-95.

Messick, S. (1995). Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *American Psychologist, 50*(9), 741-749.

Mulvey, K. L., & Irvin, M. J. (2018). Judgments and reasoning about exclusion from counter-stereotypic STEM career choices in early childhood. *Early Childhood Research Quarterly, 44*, 220-230.

National Academies of Sciences, Engineering, and Medicine. (2018). Impacts of enrollment growth on diversity in computing. In *Assessing and responding to the growth of computer science undergraduate enrollments*. National Academies Press.

Padilla, J. L., & Benítez, I. (2014). Validity evidence based on response processes. *Psicothema, 26*(1), 136-144.

Pajares, F. (1996). Self-efficacy beliefs and mathematical problem-solving of gifted students. *Contemporary Educational Psychology, 21*(4), 325-344.

Plante, I., De la Sablonnière, R., Aronson, J. M., & Théorêt, M. (2013). Gender stereotype endorsement and achievement-related outcomes: The role of competence beliefs and task values. *Contemporary Educational Psychology, 38*(3), 225-235.

Rachmatullah, A., Akram, B., Boulden, D., Mott, B., Boyer, K., Lester, J., & Wiebe, E. (2020a). Development and validation of the middle grades computer science concept inventory (MG-CSCI) assessment. *EURASIA Journal of Mathematics, Science and Technology Education, 16*(5), em1841.

Rachmatullah, A., Wiebe, E., Boulden, D., Mott, B., Boyer, K., & Lester, J. (2020b). Development and validation of the Computer Science Attitudes Scale for middle school students (MG-CS attitudes). *Computers in Human Behavior Reports, 2*, 100018.

RStudio Team (2020). *RStudio: Integrated development for R*. RStudio, PBC, Boston, MA URL http://www.rstudio.com/.

Sax, L. J., Lehman, K. J., Jacobs, J. A., Kanny, M. A., Lim, G., Monje-Paulson, L., & Zimmerman, H. B. (2017a). Anatomy of an enduring gender gap: The evolution of women's participation in computer science. *The Journal of Higher Education, 88*(2), 258-293.

Sax, L. J., Zimmerman, H. B., Blaney, J. M., Toven-Lindsey, B., & Lehman, K. (2017b). Diversifying undergraduate computer science: The role of department chairs in promoting gender and racial diversity. *Journal of Women and Minorities in Science and Engineering, 23*(2).

Schmidt, F. (2011). A theory of sex differences in technical aptitude and some supporting evidence. *Perspectives on Psychological Science, 6*, 560–573.

Scott, A., Martin, A., McAlear, F., & Koshy, S. (2017, June). Broadening participation in computing: examining experiences of girls of color. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 252-256).

Smith, K., Jagesic, S., Wyatt, J., & Ewing, M. (2018). *AP® STEM participation and postsecondary STEM outcomes: Focus on underrepresented minority, first-generation, and female students*. College Board.

Thompson, B. (2004). *Exploratory and confirmatory factor analysis*. American Psychological Association.

Tsan, J., Boyer, K. E., & Lynch, C. F. (2016, February). How early does the CS gender gap emerge? A study of collaborative problem solving in 5th grade computer science. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 388-393).

Unfried, A., Faber, M., Stanhope, D. S., & Wiebe, E. (2015). The development and validation of a measure of student attitudes toward science, technology, engineering, and math (S-STEM). *Journal of Psychoeducational Assessment, 33*(7), 622-639.

Vandenberg, J., Tsan, J., Boulden, D., Zakaria, Z., Lynch, C., Boyer, K. E., & Wiebe, E. (2020). Elementary students' understanding of CS terms. *ACM Transactions on Computing Education (TOCE), 20*(3), 1-19.

Werner, L. L., Hanks, B., & McDowell, C. (2004). Pair-programming helps female computer science students. *Journal on Educational Resources in Computing (JERIC), 4*(1), 4-es.

Wiebe, E., Unfried, A., & Faber, M. (2018). The relationship of STEM attitudes and career interest. *EURASIA Journal of Mathematics, Science and Technology Education, 14*(10).

Wigfield, A., & Eccles, J. S. (2000). Expectancy–value theory of achievement motivation. *Contemporary Educational Psychology*, *25*(1), 68-81.

Wilcox, C., & Lionelle, A. (2018, February). Quantifying the benefits of prior programming experience in an introductory computer science course. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 80-85).

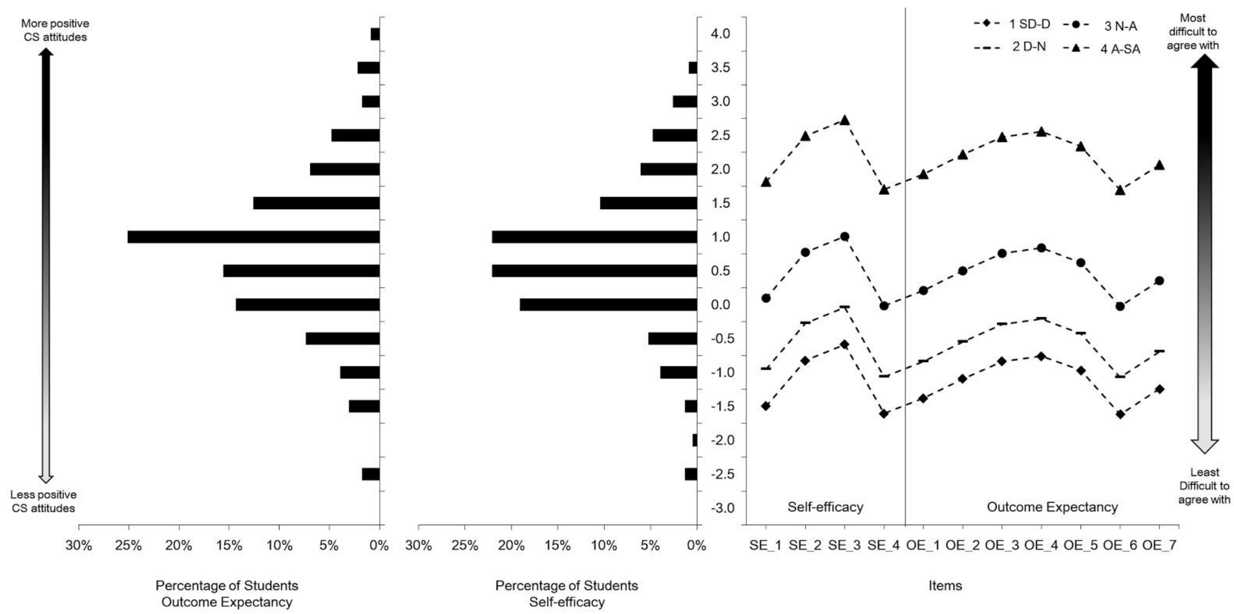
Wright, B. D., & Linacre, J. M. (1994). Reasonable mean square fit values. *Rasch Measurement Transactions, 8*(3), 370.

Yoo, M. H., Choi, J. J., Park, M. S., Chae, S. J., Kim, B. R., Son, M. H., ... & Kim, S. H. (2017). The effects of 'Silver care expert' future career-related STEAM program for the elementary students' future time perspective, career awareness and scientific attitudes. *Journal of Science Education*, *41*(1), 111-134.

Zeldin, A. L., Britner, S. L., & Pajares, F. (2008). A comparative study of the self-efficacy beliefs of successful men and women in mathematics, science, and technology careers. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching*, *45*(9), 1036-1058.

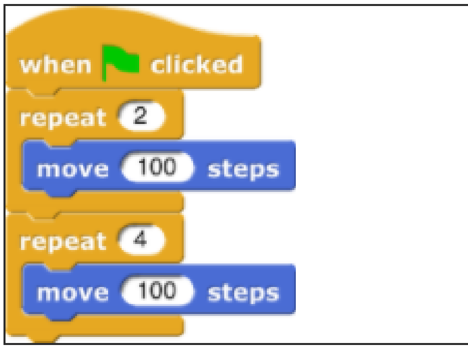
Supplemental Material

The Wright Map for The E-CSA Showing Agreement Difficulty for Each Item and Students' Attitudinal Spectrum



Note. The agreement difficulties for each item's scales on the right are represented with Thurstonian thresholds, which refer to a specific location where a student has a 50% probability of choosing a given scale or higher. Students' CS attitudes are represented with histograms on the left. When a student appears to be precisely aligned with a Thurstonian threshold, this means that the student has an equal probability of selecting the scale or option above or below the threshold. SD= Strongly Disagree, D=Disagree, N=Neutral, A=Agree, SA=Strongly Agree.)

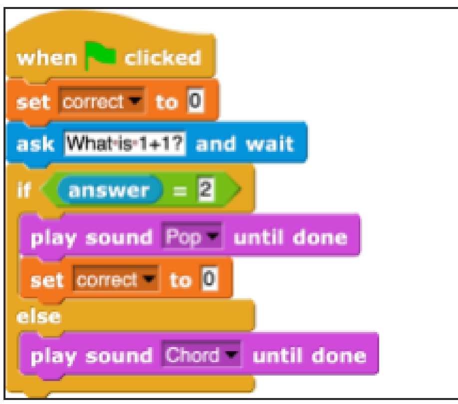
Example E-CSCA Items




```
when green flag clicked;
repeat 2 {
    move 100 steps;
}
repeat 4 {
     move 100 steps;
}
```

How many steps will the sprite move after the code runs?

- ◯ 100
- ◯ 200
- ◯ 400
- ◯ 600




```
when green flag clicked;
set (correct) to 0
ask "What is 1+1?" and wait
if ((answer) = 2) {
    play sound Pop until done;
    set (correct) to 0;
} else {
    play sound Chord until done;
}
```

What will happen if the user enters 3?

- ◯ "Pop" plays and correct is 1.
- ◯ "Chord" plays and correct is 0.
- ◯ "Pop" plays and correct is 0.
- ◯ "Chord" plays and correct is 1.

CHAPTER 4: "WE'LL FIGURE IT OUT": EXPLORING UPPER ELEMENTARY STUDENTS' COLLABORATIVE REGULATION THROUGH EPISTEMIC NETWORK ANALYSIS

Abstract

Self-efficacy is a predictor of performance. When students positively judge their competence, this likely bolsters effort and contributes to better performance. We implemented a five-week computer science intervention with upper elementary students and collected self-report data as well as transcripts of pairs of students talking while problem solving. The students' self-report data, organized by dyad, fell into three categories based on the dyad's CS self-efficacy and CS conceptual knowledge scores. Findings from within- and cross-case analyses indicate not just discursive differences, but differences in how select dyads develop a shared understanding of the task, and to whom and if they direct their help-seeking requests. Recommendations for practitioners and researchers are provided.

Introduction

Students' attitudes towards computing affect their participation in related tasks and courses, and later if they are likely to major in computer science (CS) or select a career related to computing (e.g., Cassel et al., 2007; Mitchell et al., 2009; Yardi & Bruckman, 2007). These attitudes are likely informed by students' initial experiences and exposure to computing as well as their gender (Jepsen & Perl, 2002; Mejias et al., 2019). Confidence with using a computer may be informed by students' repeated experiences in both in- and out-of-school contexts. In such instances, boys tend to report higher confidence (Beyer et al., 2002; Busch, 1995; Ogan et al., 2009); this may be due in part to their higher propensity to play video games and code recreationally (DiSalvo et al., 2014; Sevin & DeCamp, 2016).

Interest in CS varies drastically. Initial experiences may prove boring (see Goode, 2010; Moyer et al., 2018) or deleterious to one's self-efficacy (Lishinski et al., 2016). Moreover, a lack of access to early and frequent computing activities may influence students away from considering studying computing in the future (see Tran, 2018). One way to enhance interest in computing, especially among girls and historically underrepresented minorities, is through using pair programming (McDowell et al., 2006; Porter et al., 2013). This collaborative approach to learning computer programming results in higher confidence, in part, because the stereotype of solitary programmers is combatted and the two programmers engage in discussion of their thinking (Werner et al., 2004). Pair programming has mostly been studied with high school and university-aged students (e.g., Missiroli et al., 2016; Williams et al., 2000), but there is a growing interest in its application with younger students because collaborative learning strategies are regularly used in elementary classrooms (e.g., Gillies & Boyle, 2010). Pair programmers' discussions are a focal point here as we are particularly interested in how students' discourse over the course of a programming activity may illuminate individual and pair learning strategies. An individual's ability to regulate their learning significantly predicts their group's regulation (Panadero et al., 2015) and there is often a positive relationship between self-efficacy and academic regulation whereby students with high self-efficacy tend to demonstrate appropriate regulatory behaviors (Bradley et al., 2017; Pajares, 2002). Moreover, when groups utilize more regulatory strategies, they often perform better (Janssen et al. 2012).

Students' discourse gives insight into what they think, what they want, and how they make sense of tasks (Johnstone, 2017; Potter, 1998). This—students' own words while engaged in a programming activity—can be especially informative when we also consider students' self-reported attitudes toward CS and their performance on a CS assessment. This follows because of the empirical connections between a student's (Hushman & Marley, 2015) and peers' (Huang, 2017) task-related verbalizations and increases in self-efficacy, academic interest and the ability to regulate one's learning (Lee et al., 2014), and self-efficacy and performance (Brosnan, 1998; Lishinski et al., 2016). There is longstanding interest in academic self-efficacy, but far less work has been done in this area with younger students in computer science or their collaborative dynamics. The purpose of this study is to explore how students' collaborative discourse differed based on CS attitudes and performance. As the students worked in dyads, we analyzed and reported on them as a unit.

Theoretical Framework

Collaborative Regulation of Learning

Learning is often a social activity, and through talking with others, students' individual cognitive capacities are constructed and refined (Mercer et al., 1999). As such, it is important to consider how students regulate themselves and others in group learning tasks, to what extent individual regulation influences group regulation (Panadero et al., 2015), and how group processes influence an individual's acquisition of self-regulated learning skills (Hadwin & Oshige, 2011).

Research into how students collectively regulate emphasizes student interactions and fluctuations in group member influence, in particular in task performance and social processes. For example, dyadic social processes were monitored by group members more so than task processes on an inquiry-based computer-supported collaborative learning (CSCL) activity (Saab et al., 2012). This finding implies that regulating the collaborative process is paramount to, and perhaps facilitates, effective task performance. Similarly, medical students who used an interactive white board engaged in more planning and orienting which helped them establish a shared mental model early in the task, and they interacted more socially, producing more regulated discourse and ultimately making better medical decisions than those who used traditional white boards (Lajoie & Lu, 2012). In partial contrast, Janssen et al. (2012) found that groups devote approximately 35% of their time to planning and monitoring their task and 30% of their time to social processes—both creating a shared understanding and social support—however, only the regulation of social processes contributed positively to group performance.

To examine students' discursive collaborative regulation, we earlier designed, piloted, and refined a framework (Author et al., 2020b, 2021; Table 1) that included components of regulation frameworks by Hadwin et al. (2005) and Janssen et al. (2012), and a social talk framework by Kumpulainen and Mutanen (2010). The design and refinement process occurred largely because the existing frameworks included elements not relevant to the population or contexts under study.

Table 1

Collaborative Regulation of Learning Framework

Dimension	Code	Definition	Examples
Task Regulation	Planning the task	Discussion of the task, how to complete it, deciding which strategies to employ, responsibilities students will take on	Let's start by picking a background. What do we want the skit to be about?
	Monitoring task progress	Discussion of performance and progress, specific mention of strategies being used to approach the task, mentions of time	Okay, we have like five minutes left. The glide block worked better last time, so let's try that.

Table 1 (continued)

	Evaluations of task progress	Review of performance and progress, includes appraisals of task difficulty	We're never going to get all this done! That was harder than I thought it would be.
Social	Collaborative	Actively engaging with partner, attempts to maintain symmetrical contributions	What do you think the background should be? Let’s change it so she says “hello” for longer, don’t you think?
	Tutoring	Asking for or offering help/assistance	Hey, how do I add another sprite? Oh, go up to ‘operators’ and use the ‘when flag clicked’ block. Then it’ll work.
	Disagreement	Social or academic conflict	No, we’re not using that sprite. I’ll delete it if you write that in there.
	Individualistic	Working independently with no clear attempt to involve the partner	[these examples often looked like self-talk in proximity to another]
	Confusion	Failure to understand the partner or the task, often accompanied by a question	What are you talking about? That’s not what I was thinking.
	Agreement	Acknowledgements and affirmations, most often in response to a partner’s contribution	Oh yeah! Yes.

Related Works

Assessing Young Students' CS Attitudes and Performance

There is limited work on assessing upper elementary students' attitudes towards CS, coding or programming in particular. In fact, there are, to date, only three validated instruments for 4th and 5th grade students whose underlying constructs are interests and attitudes. Kong et al. (2018) developed a 15-item 5-point Likert-scale survey that queries students on programming meaningfulness and impact, and creative and programming self-efficacy. They also assessed if programming interests were related to these four factors; they found that students with more interest in programming found it to be more meaningful and impactful and had higher self-efficacies. Moreover, boys had higher interest than girls. Mason and Rich (2020) developed a 24-item 6-point Likert-scale survey that queries students on their coding confidence and interest, the perceived utility of coding, and students' perceptions of coders and social influence. They found that when students self-report high interest in coding, they have greater coding self-efficacy, supporting research in other subject areas (e.g., Grigg et al., 2018; Sheldrake, 2016). Of note, gender differences in coding attitudes were small, although statistically significant, and the authors suggest that younger students may have less exposure to gender bias in coding. This interest in young students' coding attitudes is also taken up by Vandenberg et al. (forthcoming). This 11-item 5-point Likert-scale survey, comprising the two psychological factors of self-efficacy and outcome expectancy, queries 4th and 5th grade students on their attitudes toward coding. They found that boys had statistically significantly higher CS attitudes than girls.

There is an established link between students' beliefs, such as their self-efficacy for a content area, and their performance in that area (Brosnan, 1998; Lishinski et al., 2016). In a study with six year old students, even brief experiences with programming have the potential to enhance not only their individual technology motivation and interest but such experiences also reduce students' perception that programming is only for boys (Master et al., 2017). These findings are important as they support the links between interest, self-efficacy, and intentionally designed experiences, all of which, Bandura suggested (1997), increases students' effort and contributes to higher achievement. Even less is known about how a student's interest and self-efficacy interact in a collaborative context and how these may contribute to the student's performance.

Pair Programming

In CS education, collaborative work often takes the form of pair programming. Traditional pair programming entails two students working on a single computer, each student with a designated role—the driver who has control of the input devices and the navigator who strategically guides the work (Williams et al., 2000). Both programmers are expected to talk continuously about their work, collaboratively problem solve, and to switch roles after a set amount of time or portion of the task has been completed. This pedagogical configuration has been used in industry (Canfora et al., 2007), in undergraduate classes (Williams et al., 2002), and in high school (Missiroli et al., 2016). As interest in CS education has moved to earlier grades, there is a growing interest in using pair programming with younger students (e.g., Denner et al., 2014; Shah et al., 2014; Tsan et al., 2020). Research suggests that the pair programming approach may be particularly helpful for females (Werner et al., 2004) and increases pair programmers' confidence in and enjoyment of programming (McDowell et al., 2006).

Epistemic Network Analysis

Studying the collaborative work of pair programmers inevitably means structured, systematic analysis of their discourse. Although qualitative methods of coding and analyzing dialogue have long been used in this type of research, newer methods have emerged that provide a unique and powerful insight into the discursive dynamics of dyads. Epistemic Network Analysis (ENA) is a mixed methods and data visualization technique for modeling and analyzing the connections in qualitatively tagged data. This modeling occurs by quantifying the co-occurrence of codes/tags within a conversation. This generates a weighted network of co-occurrences and related visualizations for each unit of analysis. ENA permits comparison of these resulting networks visually, statistically, and qualitatively as the approach analyzes all of the networks simultaneously. ENA was originally developed to explore the interdependence of cognition, culture, and discourse (Shaffer et al., 2016); use of the technique assumes that connections in the tagged discourse—spoken or typed—are meaningful. That is, temporally adjacent statements are likely cognitively linked. Others have used ENA to explore university students' design thinking (Arastoopour et al., 2016), surgical residents' speech and inclusion of error checklists (Ruis et al., 2018), and socioemotional group interactions in an online STEM education course (Wang et al., 2020). We used ENA with spoken discourse, using the approach detailed below.

Research Question

Guided by the literature above, we set out to answer the following question:

1. How do dyads, with different attitudes and performance scores, differ in terms of collaborative regulated discourse?

Methodology

Participants and Context

Consented participants included 60 4th and 5th grade students from a school in the southeastern United States. The school's socio-demographic data included 75%

White/Caucasian, 10% Black/African American, 5.5% Multiracial, 6.5% Latinx/Hispanic, 3% Asian, with 4% of the student population deemed low-income and 51% of the students identified as female. The education plan of the school centered around global education and awareness in addition to project-based learning. For the purposes of this study, we analyze 24 students, organized into 12 dyads; these dyads were selected based on recorded audio quality and completion of all of the instruments.

The students attended weekly technology classes and learned a series of block-based coding lessons, as taught by their technology teacher. The intervention was designed to incorporate five total lessons for each group of students. The lessons instructed students on foundational computer science concepts like how to use conditionals, loops, and variables. The teacher paired the students based on her assessment of who might work well together, and the students retained these partnerships over the duration of the study. As part of their participation in the study, all students completed several self-report surveys. The E-CSA (Vandenberg et al., forthcoming) and E-CSCA (Vandenberg et al., 2021) were administered pre intervention to assess their knowledge and attitudes coming into the study. Table 2 presents the students' pseudonyms and relevant data for the following analysis.

Instruments

Elementary-Computer Science Attitudes (E-CSA) (self-efficacy items)

This 11-item 5-point Likert scale survey intended for upper elementary use queries students on their self-efficacy and their outcome expectations for learning CS, coding more specifically, and was adopted from the middle grades version developed and validated by Rachmatullah et al. (2020). Based on a previously validated questionnaire, this version underwent revision and validation through cognitive interviewing (Author et al., 2020a) to ensure appropriate wording for young students and has undergone confirmatory factor analysis and item response theory-Rasch analysis for establishing validity and reliability (Author et al., under review A). The CS self-efficacy subscale consisted of 4 items (α = .812) and the CS outcome expectancy subscale consisted of 7 items (α = .838). For our analysis, we only used the students' answers to the subscale containing the 4 self-efficacy items. These include statements such as, "I am good at fixing code." Students were then organized into high-low categories via a median split.

Elementary-Computer Science Concepts Assessment (E-CSCA)

This 18-item multiple choice assessment intended for upper elementary use queries students on their knowledge of foundational CS concepts such as loops, conditionals, and variables and does so using mostly block-based language. Based on a validated middle grades version that assessed the same concepts (Rachmatullah et al., 2020), the elementary version's results (Vandenberg et al., 2021) indicate psychometrically sound items with no statistically significant item bias by gender or grade. We used the students' scores on the assessment to organize them into high-low categories via a median split (Table 2).

Table 2

Participant Pseudonyms and Status Indicators

Dyad #	Pseudonym	E-CSA Self-Efficacy Status	E-CSCA Status	Dyad Status
1	Melanie	high	high	High
1	Poppy	high	high	High
2	Mila	high	low	Mixed
2	Nathan	low	low	Mixed

Table 2 (continued)

3	Max	low	low	Low
3	Joshua	low	low	Low
4	Samantha	low	high	Mixed
4	Andi	high	low	Mixed
5	Rylee	low	high	Mixed
5	Amber	high	low	Mixed
6	Phoebe	low	low	Mixed
6	Kylie	high	high	Mixed
7	Emma	high	low	Mixed
7	Malachi	high	low	Mixed
8	Louis	low	high	Mixed
8	Ashley	low	high	Mixed
9	Sahil	low	high	Mixed
9	Ezra	high	low	Mixed
10	David	high	low	Mixed
10	Leo	low	low	Mixed
11	Allegra	high	low	Mixed
11	Chloe	high	low	Mixed
12	Alaina	low	low	Mixed
12	Arden	low	high	Mixed

Procedure and Analysis

Dyads were video and audio recorded each time they collaboratively programmed. We used Open Broadcaster Software (https://obsproject.com/) to align the dyads' webcam video, their audio (gathered through headsets attached to the laptop), and their screen capture. The videos, approximately 40 to 50 minutes in length, were transcribed verbatim and qualitatively tagged using the collaborative regulation of learning framework (see Table 1). The videos were selected from the second or third day of the intervention; thereby providing the students time to acquaint themselves with their partner and to learn certain CS concepts. The analyzed task most of the students were engaged in was user input, or coding a sprite to query the user and then use that information to respond. The number of discursive moves made by the dyads ranged from 431 to 1,020, with an average of 712 moves across the 12 dyads.

Each task-related utterance a student made was tagged with one code from the Task Regulation dimension, as the students demonstrated which phase of the regulation cycle they were in, and at least one code from the Social dimension, as the students used their language to communicate for specific purposes, such as to disagree or express confusion. The first author trained a second researcher on the framework and, after resolving misunderstandings of the codes, they dual coded 25% of the dataset (three transcripts/videos). An overall kappa (*k*) of .82 and agreement of .96 was reached (McHugh, 2012). The first author then proceeded to solo code the remainder of the transcripts/videos.

The qualitatively tagged transcripts were then imported into ENA along with necessary metadata, including dyad number, pseudonyms, and the individual students' scores on the E-CSA (self-efficacy) and the E-CSCA. These scores were assigned a high or low status designation determined by a median split of all the students' data (see Table 2). We then created a series of ENA models hierarchically organized by the dyads' collective performance on the two instruments.

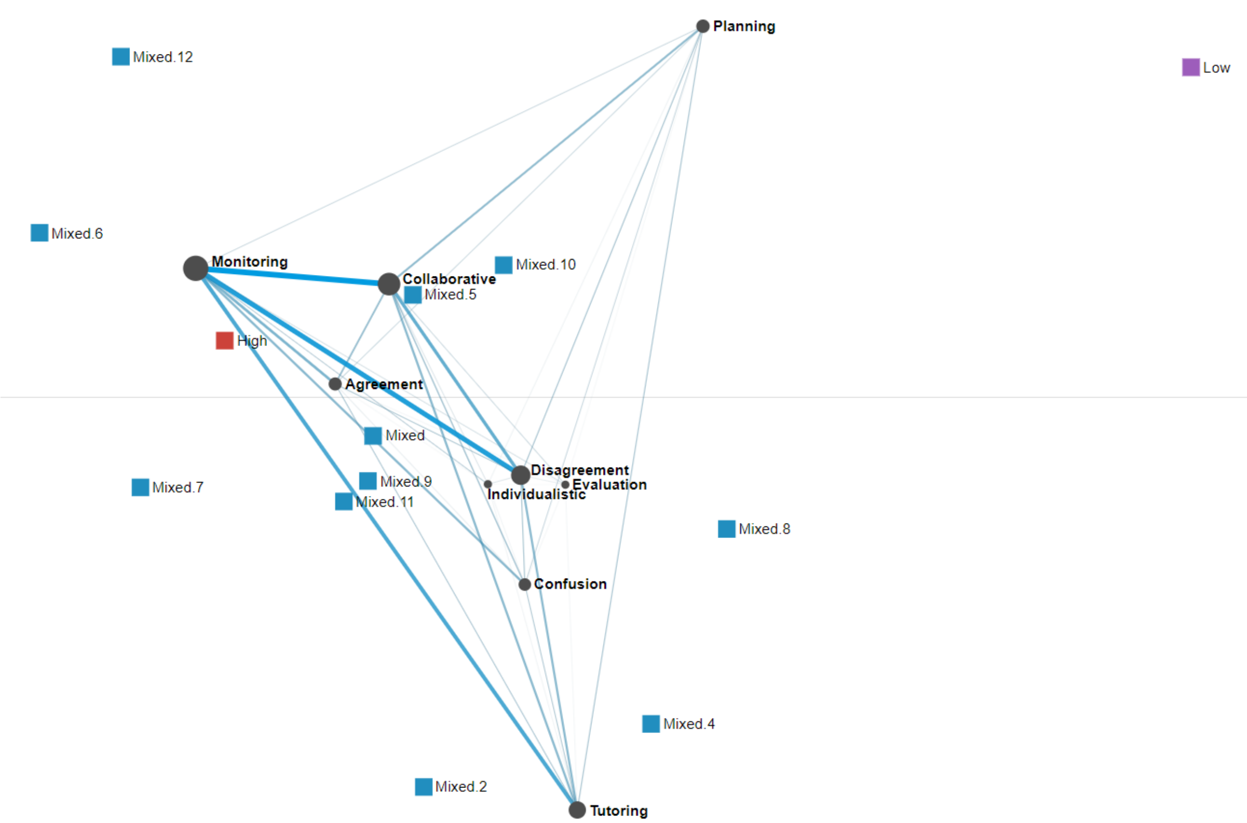
To answer our research question, we created our models using the following information. The units of analysis were all lines of data associated with a single value of Dyad Status subsetted by Dyad and by individual Student. By hierarchically organizing our units in this way, we could visually and qualitatively compare the three types of Dyad Status: Low, Mixed, and High. That is, we could see how the dyads' regulated discourse differed according to their dyadic self-efficacy and performance status. Dyad Status was determined by comparing the individual members of the dyads' CS self-efficacy status (low or high) and conceptual understanding status (low or high). Dyads where both members achieved high on both instruments were deemed High Dyad Status. Low Dyad Status was ascribed to those dyads where both members achieved low status on both instruments; Mixed Dyad Status was applied to the remaining dyads whose members had a mix of high and low scores on the two instruments.

Next, we made determinations about how ENA would make connections within the students' discourse. Temporal proximity in students' discourse likely indicates meaningful connection (Siebert-Evenstone et al., 2017). Connections within the networks are determined by creating a *window* of the co-occurrence of tags in the current statement and those within a set number of previous lines, in our case the *window* was 8 lines. Our ENA model included the following tagged collaborative regulation categories (from Table 1): Planning, Monitoring, Evaluation, Collaborative, Agreement, Tutoring, Disagreement, Confusion, and Individualistic. In each network model, each node represents a collaborative regulation category. A dimensional reduction via a single value decomposition (SVD) algorithm was used to rotate the model, similar to what occurs in principal components analysis, so that the x-axis explains the greatest variance among the units and the y-axis explains the second greatest variance (Arastoopour et al., 2016).

Models were analyzed according to the node size, edge (or line) thickness, and node placement (see Figure 1). Node size indicates the frequency of the corresponding collaborative regulation category that occurred relative to all category co-occurrences. Edge thickness, which appears both as width and color intensity, indicates the relative frequency of the connected collaborative regulation categories, or nodes. ENA places nodes using an optimization routine so that the centroid for any unit under analysis is as close as possible to the point in projected ENA space (Shaffer et al., 2016). A centroid is the mean for a network model and is represented as a circle or square. In our models, the dyad centroids are represented as unconnected squares, with red denoting high, blue denoting mixed, and purple denoting low, and represent the mean of all dyads within that group.

Figure 1

ENA Example Model



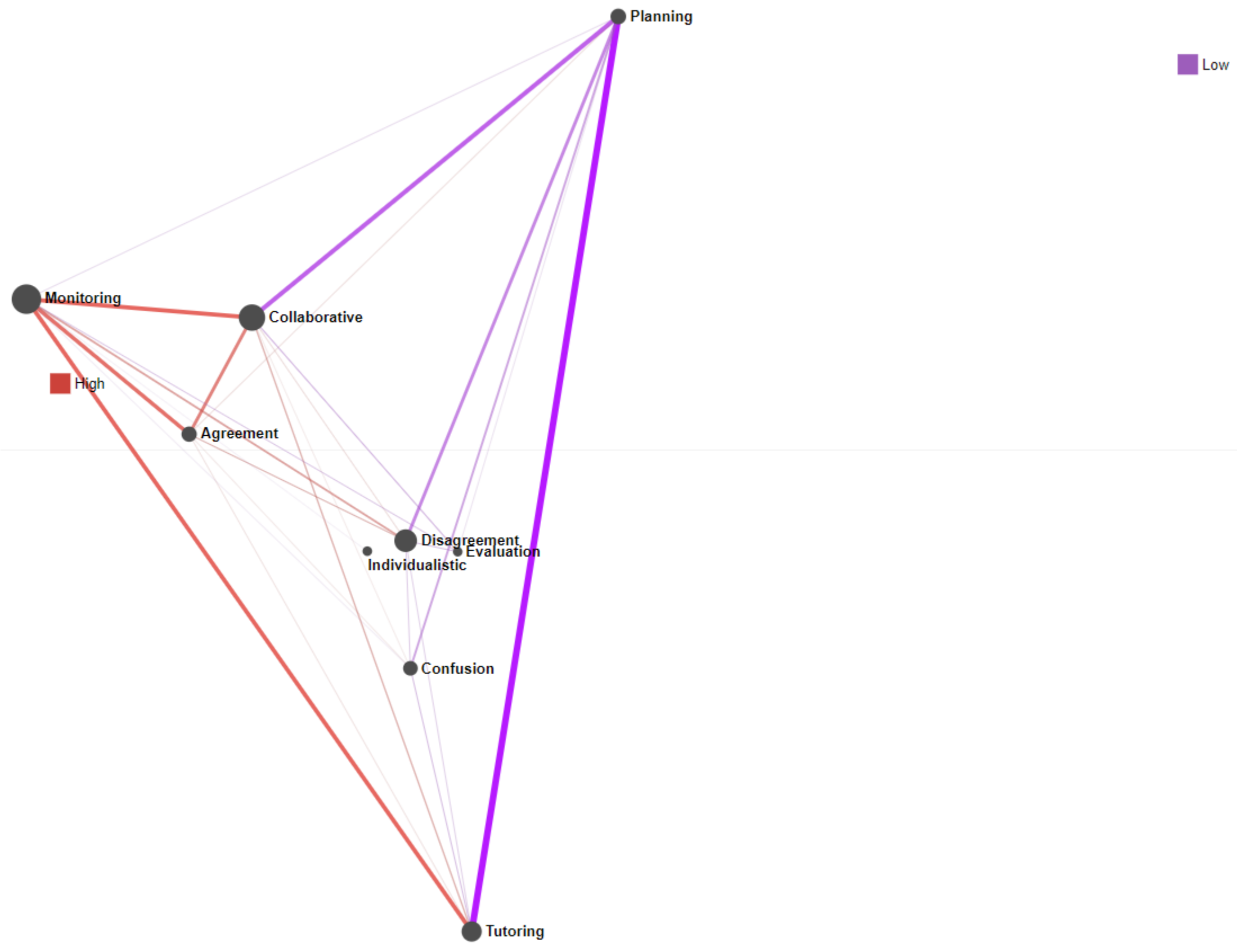
Our use of median splits with the E-CSA and the E-CSCA permitted us to cleave off extreme cases. These—the high and low groups—in many ways reveal discursive patterns we anticipated; therefore, we start our analysis there before more closely examining the mixed cases to see what thematic groupings emerge. Prior work infers that these mixed dyads may provide more insight as to how self-efficacy and prior knowledge affects discursive patterns (cf., Denner et al., 2014; Mohammed, 2019).

Findings and Discussion

The nodes in ENA permit us to compare several models and to ascribe meaning to where dyads' centroids reside in the model, relative to the nodes. Figure 2 displays singular, extreme cases of High and Low dyads. That is to say, one dyad had each member score as high on both instruments, noted as the red High square in the figure, and one dyad had each member score as low on both instruments, noted as the purple Low square in the figure. We begin our findings by providing a within-case report for each of these extreme cases.

Figure 2

ENA Difference Model of High and Low Dyads



Note: Red denotes the High dyad, purple denotes the Low dyad.

Extreme Cases: High to Low

High: Dyad 1

Dyad 1 students, Poppy and Melanie, collectively uttered more co-occurring statements tagged as Monitoring and Collaborative, Monitoring and Agreement, Collaborative and Agreement, and Monitoring and Tutoring. Collaborative regulation researchers maintain that students' use of task-regulating behaviors (e.g., monitoring) contributes to effective collaboration

(De Jong et al., 2005; Janssen et al., 2012). Excerpts from the students' work demonstrate how the two smoothly move from collaboratively agreeing on their problem-solving approach to questioning the other's strategy to talking through their individual desires to resolving minor disagreements, such as in the excerpt below.

Table 3

High Dyad 1: Excerpt 1

Student (Driver or Navigator)	Utterance
Melanie (N)	Oh, we need to put a space in, in front of... I wonder if we should do a say [block]... Wait, why'd you put that?
Poppy (D)	Oh.
Melanie (N)	Back space and then space. I wonder if we should do that on another line so that, like another say, so that it's not like, so awkward 'cause it's like all in one, you know?
Poppy (D)	[reading the code] What's your name? PM, enter. Hello, PM. Nice to meet you.
Melanie (N)	I think we should do like-
Poppy (D)	No, I think, I think, I think it's good like this 'cause it, 'cause-
Melanie (N)	But it's all jumbled, I feel. I feel like it could just be a little neater.
Poppy (D)	Hm.
Melanie (N)	What do you think?
Poppy (D)	Should we do it? I feel like we should-
Melanie (N)	Can we just try it and then-
Poppy (D)	Hello, PM, then it would go off. Then it would say nice to meet- [runs the code]
Melanie (N)	Yeah.
Poppy (D)	... you. Yeah, I guess that would work.

In this excerpt, Melanie, who is navigating, encourages Poppy, the driver, to clean up the code so it appears neater. Poppy initially resists, but after Melanie asks to "just try it," Poppy realizes that the "jumbled" code blocks are not simply messy looking but that they are not permitting the program to run how the girls desired.

Grounding, or seeking shared understanding of the task goal or next steps, is essential for collaboration and to prevent unproductive conflict (Erkens et al., 2006; Jeong & Hmelo-Silver, 2016). In Poppy and Melanie's discourse, grounding occurs through their use of Tutoring and Agreement statements, in particular. The Tutoring statements, as noted below, were often simple requests for clarification and task-related assistance so the two could remain grounded.

Table 4

High Dyad 1: Excerpt 2

Student (Driver or Navigator)	Utterance
Melanie (N)	So we need to use a loop.
Poppy (D)	But which?
Melanie (N)	Oh this one- *points at the screen* If space key pressed-
Poppy (D)	Okay.
Melanie (N)	This is-
Poppy (D)	So, if space key pressed, this happens. And then you need to put that in a loop.
Melanie (N)	Space key pressed.
Poppy (D)	So pick three letters. How about A, M, and T?
Melanie (N)	Okay.
...	*… the girls debate which fruits to represent the letters during which time Poppy references class time ending soon*

Table 4 (continued)

Melanie (N)	But we need to work, but, okay, so we need to put this in a loop but we need to make it so this changes every time.
Poppy (D)	Eh, that's hard.
Melanie (N)	But for now, let's try it.
Poppy (D)	Say "mango" for 2 seconds
Melanie (N)	Wait. It needs to like ask [the user] something-

In this excerpt, the girls were tasked with building code such that a user would be given options of letters to select and, upon selecting one, a written response would appear in addition to an image, in this case fruit. Poppy is the driver again and receives tutoring assistance from Melanie in the form reminders of needing to use a loop, which one to use, and why that loop block is necessary.

Low: Dyad 3

Dyad 3 students, Max and Joshua, offered more co-occurring statements that mostly included Planning. That is, Planning and Collaborative, Planning and Disagreement, Planning and Confusion, and Planning and Tutoring. Collaborative regulation research often reports that students rarely plan (Järvelä & Hadwin, 2013), so these consistent co-occurring statements that include Planning are intriguing. However, upon closer examination of the dyad's transcripts, in conjunction with watching their video, the boys seldom progressed beyond the planning stage of the task as they intermittently sang songs, engaged in off-task conversations about popular YouTube videos, and, after discovering the decibel scale for the research audio collection, made loud noises to get the scale to "go red." They only focused on their coding task when they saw an adult nearby or when their teacher sat and took them through the lesson step-by-step.

The boys' use of Tutoring statements largely revolved around exclamations for help (ie., "Max! Help me!") and questions such as "Max look, I need help, where do we go now?" and "Can you help me?" A brief excerpt that includes Planning, Tutoring, Confusion, and Disagreement statements follows.

Table 5

Low Dyad 3: Excerpt 1

Student (Driver or Navigator)	Utterance
Joshua (D)	Max! Help me!
Max (N)	Dude, you're the driver.
Joshua (D)	I don't know how to do this.
Max (N)	Wait. What are we supposed to do?
Joshua (D)	*Reading the directions on the screen* Create a program that takes- that takes...
Max (N)	Okay.
Joshua (D)	*Reading the directions on the screen* … in your user's name and greets them.
Max (N)	So the first thing we do- You go to the costumes
Joshua (D)	Wait. I don't know. Is it-?
Max (N)	Today… *Begins talking about a YouTube challenge video*

Following this excerpt, the boys engage in off-task banter interrupted twice by Joshua asking Max for help. Joshua drags one block to the scripting area during this five-minute time period, after which Joshua states, "Okay, we're going to start." The teacher appears in the video and guides the two toward selecting the next block which is intended to ask the user their name. The boys then begin to chant "what's your name?" and make nonsense sounds repeatedly.

The two only engaged in Monitoring behaviors once during the almost 42-minute coding session. That excerpt appears below, with co-occurring Confusion and Disagreement utterances. It is important to note that Max, as driver, is the only one interacting with the programming environment at the beginning of this excerpt.

Table 6

Low Dyad 3: Excerpt 2

Student (Driver or Navigator)	Utterance
Max (D)	This is- I don't get this. So I put all of these in the if [block]?
Joshua (N)	*Inaudible (not wearing headset)*
Max (D)	So all of them in? Oh- just the say [block].
Joshua (N)	*Inaudible (not wearing headset)*
Max (D)	If-
Joshua (D)	*Shifts laptop toward himself, becomes audible, and assumes the Driver role* I know, I know.
Max (N)	Answer is right. Answer is right. Answer.
Joshua (D)	*Pretends to remove the codeblocks Max just added* No, no dang.
Max (N)	Dude. Stop. Dude.
Joshua (D)	What, dang? Just kidding.
Max (N)	I'm so confused.
Joshua (D)	*Opens a new window and Google searches "I'm so confused" and both boys laugh*

For the remaining 20 minutes of the coding session, the boys worked without adult assistance only twice; one time they worked to change what the sprite said and the other time they changed the sprite's size. However, three times, adults intervened for a total of 9 minutes and 37 seconds.

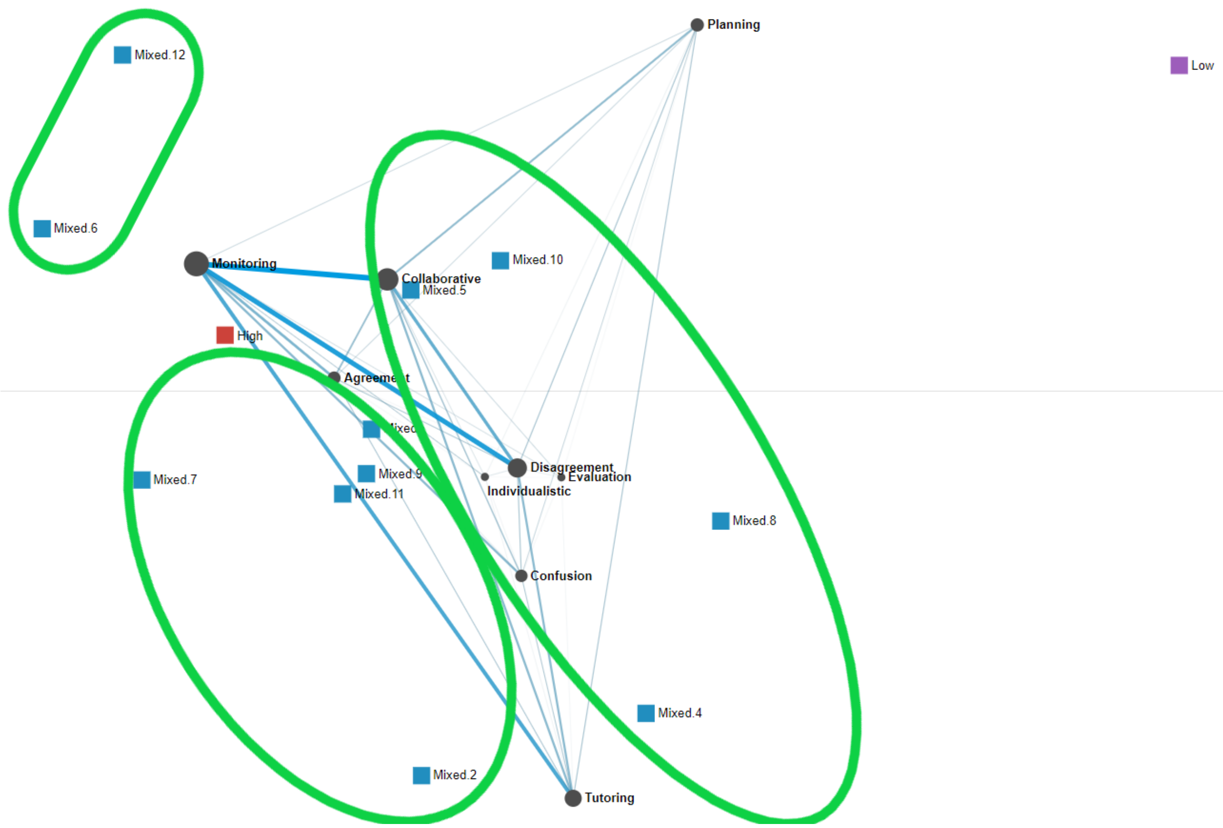
Of the five tasks the students were expected to complete, the boys correctly completed two and both were done with adult assistance.

Mixed Status Dyads

The remaining 10 dyads comprise the Mixed Status. Noted previously, we used a more grounded approach with the analysis of these dyads as there is minimal literature to guide how young students' individual CS self-efficacies and performance may influence their collaborative and regulated discourse. Therefore, we used ENA to qualitatively cluster the dyads to highlight notable attributes within the Mixed Status. The following ENA model (Figure 3) displays these dyads, from which we pull representative cases. Dyads 6 and 12 qualitatively cluster together as they both utilized mostly Monitoring and Collaborative statements. Dyads 2, 7, 9, and 11 form the next group as they uttered more Disagreement and Individualistic statements than the other dyads. The final cluster includes 4, 5, 8, and 10 who used Disagreement, Confusion, and Tutoring more so than the other dyads.

Figure 3

ENA Model of Mixed Dyads



Note: Blue denotes Mixed dyads. The blue squares are the individual dyad centroids, or mean, for the network. The green ovals indicate the qualitative clusters.

Mixed: Dyad 6 (Collaborative)

Phoebe and Kylie are dyad 6 students. Phoebe scored low on both the self-efficacy and CS conceptual knowledge measures, whereas Kylie scored high on both. In the following excerpt, the girls had just completed the expected task and are talking through how to add more to the code. Kylie is the driver and Phoebe, as navigator, both make recommendations on the next steps and echoes Kylie's contributions.

Table 7

Mixed Dyad 6: Excerpt 1

Student (Driver or Navigator)	Utterance
Kylie (D)	We did it!
Phoebe (N)	Ah! Oh, and m- maybe we could say, a- ask it to say, "What's your name?"
Kylie (D)	Yeah, but that's what we got-
Phoebe (N)	No. Like, … make it s- say a little bit more so we can show [the teacher].
Kylie (D)	Oh.
Phoebe (N)	Right. "What's your favorite food?"
Kylie (D)	Oh. Wait, yeah. Ask, ask. There it is. Well, sensing.
Phoebe (N)	That's not sensing, right? "What's your favorite food?" What's... Your... Favorite... Food. Then you go to like...
Kylie (D)	And then go [to] sound.
Phoebe (N)	...sound
Kylie (D)	Looks.
Phoebe (N)	Looks. And then... Say, "Mm, I like that food too!" Something like that.

Collaboration does not always imply agreement, of course. Kylie and Phoebe experienced instances of disagreement, as demonstrated in the following excerpt. It is noteworthy that their disagreements do not prevent them from working productively, however.

Table 8

Mixed Dyad 6: Excerpt 2

Student (Driver or Navigator)	Utterance
Kylie (D)	But, no, I- I don't think that's what we're supposed to do. I don't think-
Phoebe (N)	I know, we- we'll figure it out.
Kylie (D)	Thing is, I don't know how to do it.
Phoebe (N)	Then, I know how to do it though.
Kylie (D)	Maybe I, maybe we're-
Phoebe (N)	So write answer there, we put answer there. *Points at screen*
Kylie (D)	But [raises her hand for adult help]-
Phoebe (N)	Kylie, and then if we get it wrong, there's time. Then we can ask for help. Please.
Kylie (D)	No, we did get it wrong though Phoebe-
Phoebe (N)	No, because you didn't let me finish.
	Teacher arrives and asks Phoebe, as navigator, what she wanted to see happen. Phoebe explains and the teacher confirms she was correct, but the girls need to use a certain operator
Phoebe (D)	Yeah, equals. If answer equals...
Kylie (N)	40.
Phoebe (D)	40. Sorry, Kylie. *Leans over and side hugs Kylie.*

Mixed: Dyad 9 (Disagreement and Individualistic)

Dyad 9 includes Sahil and Ezra. Sahil scored low on self-efficacy and high on CS conceptual knowledge, whereas Ezra scored high on self-efficacy and low on CS conceptual knowledge. Sahil and his partner, Ezra, however, do not experience the same level of understanding, symmetry, and collaboration as Phoebe and Kylie in dyad 6. Of the four dyads in this cluster, dyad 9 is closer to the Disagreement and Individualistic nodes. Regarding disagreement-coded utterances, it is important to recall that these can be both simple "no" statements as well as statements reflective of conflict, such as name calling. In the following excerpt, Sahil's partner calls him a name and when Sahil made a suggestion for code edits, Ezra countered without any discussion or justification.

Table 9

Mixed Dyad 9: Excerpt 1

Student (Driver or Navigator)	Utterance
Ezra (D)	I'm driver, you idiot.
Sahil (N)	Make him 350.
Ezra (D)	Okay, one minute. Okay.
Sahil (N)	Um (laughs) He's having fun. Yeah, that good. That's good.
Ezra (D)	No, no, no. 250. 250. Yeah, that's perfect.

Sahil and Ezra also engaged in distinctive types of tutoring with one another. In some instances, the boys simply used short demands as offers of help, such as "Move!" when they desired their partner to use the move block. In other instances, one would attempt to tutor the other, generally, Sahil offering support for Ezra, resulting in a disagreement, giving rise to Ezra physically disengaging from the task (ie., leaving the space or leaning back and talking to a nearby friend) and Sahil verbalizing his work Individualistically. One of these instances follows.

Table 10

Mixed Dyad 9: Excerpt 2

Student (Driver or Navigator)	Utterance
Sahil (N)	Oh you're the first driver, it says the first driver is student 4 so that's you.
Ezra (D)	What do I do?
Sahil (N)	Um, what do you have to do? Oh yeah we have to make a thing last for 20 seconds. So first of all, we're not using that [turtle] sprite, right? *Sees Ezra opening a tab*: Ay no. No no no no no.
Ezra (D)	Idiot.
Sahil (N)	No not me, you're on the wrong thing. You're on the wrong thing. You have to close that. You have to first click the Sprite. All right, whatever then just do it. Save to cloud. Um, look at all the [sprites]-
Ezra (D)	Ew, he looks so retarded (laughs). *Looks behind him*: I'll be right back. *Ezra is off screen for 22 seconds.* We need this one. Import it. (laughs) Oh that's... oh.
Sahil (D)	*Assumes Driver role in Ezra's absence* No, you have to find two, she said you had to have two Sprites so we have to find another guy that looks like that. *Ezra leaves again and for just over two minutes, Sahil scans sprites, adds several, and learns how to change their orientation on the stage. Ezra watches for an additional minute, talking about the headsets they are wearing.*
Ezra (D)	*Takes back the laptop and assumes Driver role* I get to drive first you idiot.

This type of exchange is fairly common with these boys. They did not regularly follow the driver-navigator prompts, but more fluidly asserted control over the laptop, often with little discussion. Ezra was frequently interested in sharing the dyad's progress with a friend in another group, often getting up and visiting that friend to tell him what he and Sahil had done. This left Sahil alone to make coding changes or to wait until Ezra returned.

Mixed: Dyad 8 (Disagreement, Confusion, and Tutoring)

Louis and Ashley comprise dyad 8. They both scored low on self-efficacy and high on CS conceptual knowledge. Together they had an unfortunate, incorrect, and rather common misunderstanding of the roles the driver and navigator were expected to enact. Both assumed, and regularly verbalized, that the navigator's role was to tell the driver what to do. This inevitably led to Tutoring requests, perhaps fueled by Confusion, and Disagreement. Despite having higher CS conceptual knowledge, they often did not have the language to describe what they wanted their partner to do; in this case, it may be that teaching or Tutoring a peer was cognitively taxing. An excerpt follows.

Table 11

Mixed Dyad 8: Excerpt 1

Student (Driver or Navigator)	Utterance
Louis (D)	Tell me what to do.
Ashley (N)	Get a block.
Louis (D)	What block?!
Ashley (N)	Go to operations. Then you hit the equals sign.
Louis (D)	Equals sign.
Ashley (N)	Then bring it out. No, it doesn't go in there, Louis. Make it true. No you have to do equals party, you have to do equals party.
Louis (D)	I don't...you're not giving specific instructions.
Ashley (N)	Okay, you only need to modify the sprite's code to different animations for each room. Sprite.
Louis (D)	Alright, so.
Ashley (N)	So he has to have, she has to have a different sprite.
Louis (D)	Alright. So, go to costumes.

Table 11 (continued)

Ashley (N)	Mm-hmm (affirmative) No...
Louis (D)	Well, you're not telling me what to do, that's your job, do your job! If you want this...
Ashley (N)	Go to costumes.
Louis (D)	Well, you just groaned when I tried to do that.
Ashley (N)	No, you don't just have to switch, you have to create code to make it switch first.
Louis (D)	You told me to go to costumes!
Ashley (N)	(groan)
Louis (D)	(sigh) So, what do you want me to do? You're not telling me at all. You told me to go to costumes, and then you tell me to get out of costumes.
Ashley (N)	Push close. Okay, go to control. Go. To. Um, no. Go to if. No, you, this, I pointed to that one.

Cross Case Discussion

Despite differences in the students' CS self-efficacy and conceptual knowledge, there remain similarities and differences in the ways they approached collaboration—as evidenced through their discourse—and the verbalization of their knowledge.

Grounding

Grounding, or developing a shared understanding, occurred in varied ways across the dyads. Dyad 1 (Poppy and Melanie) readily engaged in grounding through question-asking. A brief excerpt follows. Here the girls are deciding how their sprite will ask the user their name.

Table 12

Dyad 1 Grounding Excerpt

Student (Driver or Navigator)	Utterance
Melanie (D)	And then how do we want to say that, what's your name?
Poppy (N)	Just like hello.
Melanie (D)	Well no, but like how do we want to say, "Hi"?
Poppy (N)	Oh, do you wanna be like French or-
Melanie (D)	You can say, "Bonjour". We can say, "Hi, Hello, Hola." What do you wanna do?
Poppy (N)	Hm. I'm not sure. Maybe hola. Oh, or we could do all of them.

Such incremental grounding occurs when the participants must determine each other's meaning, understanding, or expectations in order to move forward together (Brennan, 1998). In this case, the girls talked through how they envisioned the sprite saying hello.

Kylie and Phoebe (dyad 6) approached grounding differently. Whereas Poppy and Melanie achieved this through a symmetrical question asking and answering exchange, Kylie and Phoebe's interactions were more asymmetrical, with Phoebe often deferring to Kylie's knowledge and Kylie asking for adult help when she could not complete a task. The grounding these girls achieved occurred through simple acknowledgements (ie., "yeah" and "uh huh") and echoing of one another's statements (e.g., "Go to costumes" followed by the other saying "costumes"). These repetitions and utterances of support are common within girl-only groups (Colfer, 2011), especially as ways of establishing and maintaining connections with one another.

Dyad 8, Louis and Ashley, used some similar repetitive grounding utterances as dyad 6 (ie., "delete" followed by the other saying "delete"). These students heavily relied on each other as administrators of their actions; that is, when Louis was driving, he expected Ashley to verbally navigate his every action, and she expected the same of him. This was a unique form of grounding, in which both students freely relinquished decision making control over to their partner at certain times of the activity, and readily assumed this responsibility when necessary.

The remaining two dyads did not engage in verbal efforts to establish a shared understanding of the task. Dyad 3 (Max and Joshua) were regularly off-task and worked on task only when an adult was near or sitting with them, at which point their discourse shifted to focus more so on the teacher; that is, the student who was driving and the teacher talked to one another, often leaving the navigator silent. Dyad 9 (Sahil and Ezra) also did not actively seek common ground with one another. Ezra was often physically absent from the task (ie., off the camera) and regularly called his partner an idiot. Colfer (2011) found that all boy groups tend to engage in disputational talk, marked by assertive and critical demands. Moreover, she found that these groups were highly individualistic as seen in the number of I-statements. Dyad 9's excerpts demonstrate such I-statement usage, discouraging remarks, and individual decision making.

Help Seeking

Across the five dyads examined here, different types of help-seeking behaviors were also observed. When students are cognizant of what they know, have an awareness of what they can accomplish on their own, and know when and how to seek out a more knowledgeable other for assistance, they are thought to have adaptive help-seeking strategies (Newman, 2002). The dyads examined here demonstrate this capability differently.

Dyad 1 (Poppy and Melanie) sought adult assistance twice during the coding activity. The first time, the girls could not find a certain block, but by the time the teacher appeared, they had found the correct block and had moved on. The other time, a researcher was walking by as

Melanie said "we don't understand it really" which prompted the researcher to stop and assist. Several other times, the teacher and a researcher appear on screen to check on the girls' work. Otherwise, the girls make use of each other's knowledge, sometimes acknowledging that they do not know the answer or the correct next step but moving ahead with "let's just try it."

As noted above, dyad 8 (Louis and Ashley) relied heavily on each other's knowledge. Interestingly, when one drove, he/she rarely challenged the other's requests but instead executed whatever the navigator directed. These exchanges were punctuated by requests for more information or clarification, but the students infrequently disagreed over the demands made by their partner. In this dyad, the partner's knowledge was virtually absolute. The only instance contrary to this occurred as Ashley drove and Louis's code did not produce the outcome expected to which Louis responded "well, that's not my fault" and Ashley raised her hand for adult help. Why she did not attempt to fix the code herself, acting as both driver and navigator, we do not know.

Phoebe and Kylie in dyad 6 began their interaction much like Poppy and Melanie. However, a shift of some sort occurred when Melanie no longer felt able to complete the task. At that point, Phoebe verbalized that she knew how to do it. Melanie raised her hand for adult assistance, regardless. We do not have insight into why Melanie did not permit Phoebe to make the coding edits she desired; perhaps Melanie, having acted as the dyad's more knowledgeable partner, did not trust Phoebe's actions. It is worth noting that after the teacher confirmed that Phoebe was correct, Phoebe uttered more demands (ie., "No, wait. Test it first.")

It is difficult to state that the remaining two dyads demonstrated *adaptive* help-seeking behaviors. However, there were instances in which the boys sought their partner's assistance. Joshua asked or demanded Max (dyad 3) help him a total of 10 times during the coding activity.

No such appeals came from Max to Joshua, and in response to the 10 requests for assistance, Max either ignored them and changed the subject (e.g., "What's up guys. Back to the kitchen. We're always cooking some videos.") or deflected responsibility (e.g., "Dude, you're the driver."). Neither of the boys explicitly asked for teacher assistance, but it was given when it became apparent that the boys were struggling. Dyad 9 (Sahil and Ezra) seemed to struggle with focused attention and collaboration. Noted previously, Ezra was often out of his seat, visiting a friend's table. Ezra in particular, and Sahil to a lesser extent, seemed interested in getting this third student's (Nathan) attention and approval of their work (e.g., "Hey Nathan! Look, look at this!"). Sahil and Ezra rarely asked each other questions (e.g., "What do I do?") nor for adult assistance. The latter was given with regularity, and often in response to Ezra's absence. That is, the teacher would see Sahil alone and call Ezra back to their table, offer a brief intervention, and leave again.

Conclusions, Limitations, and Future Work

Self-efficacy is a predictor of performance, with positive estimates of one's competence likely bolstering effort and contributing to higher achievement (Bandura, 1997). Maladaptive, or inaccurate, estimates of one's efficacy, however, can be problematic as they may lead to a lack of awareness of when to seek help and when to apply appropriate learning strategies (Bandura, 1989).

Regarding students' assessment of their capabilities to successfully complete certain CS-specific actions, we were struck by the finding that the only High group was made up of girls and the only Low group was made up of boys. Although these pairings work against the prevailing literature that boys are generally overconfident in their assessment of their capabilities (Beyer et al., 2003; Cheryan & Plaut, 2010) and that they tend to perform better and report more accurately their performance than girls in CS (Kallia & Sentance, 2018), the small sample of only two dyads prevents us from drawing conclusions. We do believe this is worth exploring further with this young age group, however.

It is essential for students to appropriately assess their abilities as inaccurate understandings can prevent students from asking for assistance. Computer science is one subject area that can easily provide such feedback to students as they can run their code and immediately know the accuracy of their work. Future work may consider expanding on Roll et al.'s (2011) finding that a self-assessment tutor improves students' accuracy. Similarly, future efforts in learning analytics may consider the use of on-screen prompts to guide students' collaborative discourse.

Efforts to improve low student self-efficacy are varied. Crippen and Earl (2007) found that students in an online Chemistry class had improved self-efficacy and performance when provided a worked example and the requirement to self-explain. This type of intervention would be straightforward and appropriate to integrate into a CS setting, especially one that uses pair programming where students are expected to talk through their thinking. The younger the students are in a coding education intervention, the more likely they are to report statistically significant differences in self-efficacy (Okal et al., 2020). Therefore, the earlier students are exposed to programming, the more their self-efficacy will be positively affected; programming experience helps build programming self-efficacy (Mazman & Altun, 2013; Resnick et al., 2009). The students in our study all participated in the same weekly intervention at school, but may have had different at-home and out-of-school experiences that influenced their interest in and self-efficacy for CS.

Lastly, personality differences may have influenced not only the individual's performance on the CS conceptual knowledge assessment, ways he or she self-assessed their CS efficacy, and their experience while collaboratively coding, but this may have affected the pair's problem-solving ability. In particular, Pietarinen et al. (2018) found that if students report feeling confident, they were more likely to actively participate, collaborate, and support their group members than if they were feeling less confident, or insecure. We believe that some of the differences we saw in our groups likely hinge on the individual students' belief in their ability to complete the CS work as it is in tension with the belief of that of their partner. In other words, one student's high self-efficacy might not be enough to overcome the lack of support and disinterest in the task a partner may have offered. As such, pairing students by similar collaboration interests or self-efficacy may be a consideration for future research (see Campe et al., 2019).

Our study was limited in sample size and diverse socio-demographic characteristics and our findings need to be interpreted with respect to these limitations. Future research could utilize this analysis approach with a larger and more diverse sample. Moreover, analyses around students' prior experiences in programming are important to incorporate. Lastly, we were unable to gather complete post-intervention data due to the COVID-19 pandemic; this study would have benefited from a thorough pre-post analysis of both the students' CS Attitudes and their CS conceptual knowledge.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No <redacted>. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Arastoopour, G., Shaffer, D. W., Swiecki, Z., Ruis, A. R., & Chesler, N. C. (2016). Teaching and assessing engineering design thinking with virtual internships and epistemic network analysis. *International Journal of Engineering Education, 32*(2).

Author et al. (in review A).

Author et al. (in review B).

Author et al. (2020a).

Author et al. (2020b).

Author et al. (2021).

Bandura, A. (1997). Self-efficacy: The exercise of control. New York: Freeman.

Beyer, S. (1999). The accuracy of academic gender stereotypes. Sex Roles, 40(9-10), 787-813.

S. Beyer, M. Chavez, and K. Rynes. Gender differences in attitudes toward confidence in computer science. In *The Annual Meeting of the Midwestern Psychological Association*, 2002.

Beyer, S., Rynes, K., Perrault, J., Hay, K., & Haller, S. (2003, January). Gender differences in computer science students. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (pp. 49-53).

Beyer, S. (2014). Why are women underrepresented in computer science? Gender differences in stereotypes, self-efficacy, values, and interests and predictors of future CS course-taking and grades. *Computer Science Education, 24*(2-3), 153-192

Bollin A., Kesselbacher M., Mößlacher C. (2020) Ready for computing science? A closer look at personality, interests and self-concept of girls and boys at secondary level. In Kori

K., Laanpere M. (Eds.). *Informatics in Schools. Engaging Learners in Computational Thinking*. ISSEP 2020. Lecture Notes in Computer Science, vol 12518. Springer, Cham.

Bradley, R. L., Browne, B. L., & Kelley, H. M. (2017). Examining the influence of self-efficacy and self-regulation in online learning. *College Student Journal, 51*(4), 518-530.

Brennan, S. E. (1998). The grounding problem in conversations with and through computers. In S.R. Fussel and R.J. Kreuz (Eds.) *Social and Cognitive Approaches to Interpersonal Communication*. Mahwah, NJ: Erlbaum, pp. 201–225.

Brosnan, M. J. (1998). The impact of computer anxiety and self‐efficacy upon performance. *Journal of Computer Assisted Learning*, *14*(3), 223-234.

Campe, S., Green, E. & Denner, J. (2019). *K-12 Pair Programming Toolkit*. ETR, Scotts Valley, CA.

Canfora, G., Cimitile, A., Garcia, F., Piattini, M., & Visaggio, C. A. (2007). Evaluating performances of pair designing in industry. *Journal of Systems and Software, 80*(8), 1317–1327.

Cassel, L., McGettrick, A., Guzdial, M., & Roberts, E. (2007). The current crisis in computing: what are the real issues?. *ACM SIGCSE Bulletin*, *39*(1), 329-330.

Cheryan, S., & Plaut, V. C. (2010). Explaining underrepresentation: A theory of precluded interest. *Sex Roles*, *63*(7-8), 475-488.

Cheryan, S., Plaut, V. C., Davies, P. G., & Steele, C. M. (2009). Ambient belonging: how stereotypical cues impact gender participation in computer science. *Journal of Personality and Social Psychology*, *97*(6), 1045.

Colfer, C. (2011). What types of talk are boys and girls using when engaged in a collaborative design and technology task? In K. Stables, C. Benson & M. de Vries (Eds.*), PATT-25*

Conference, perspectives on learning in design and technology education, London, UK, 113–128.

Crippen, K. J., & Earl, B. L. (2007). The impact of web-based worked examples and self-explanation on performance, problem solving, and self-efficacy. *Computers & Education*, *49*(3), 809-821.

Denner, J., Werner, L., Campe, S., & Ortiz, E. (2014). Pair programming: Under what conditions is it advantageous for middle school students? *Journal of Research on Technology in Education*, *46*(3), 277–296.

DeVellis, R. F. (2017). *Scale development: Theory and applications* (4th ed.). Los Angeles, CA: Sage.

DiSalvo, B., Guzdial, M., Bruckman, A., & McKlin, T. (2014). Saving face while geeking out: Video game testing as a justification for learning computer science. *Journal of the Learning Sciences*, *23*(3), 272-315.

Erkens, G., Prangsma, M., & Jaspers, J. (2006). Planning and coordinating activities in collaborative learning. In A. M. O'Donnell, C. E. Hmelo-Silver, & G. Erkens (Eds.), *Collaborative learning, reasoning, and technology* (pp. 233–263). Mahwah: Erlbaum.

Gillies, R. M., & Boyle, M. (2010). Teachers' reflections on cooperative learning: Issues of implementation. *Teaching and Teacher Education*, *26*(4), 933-940.

Goode, J. (2010, March). Connecting k-16 curriculum & policy: making computer science engaging, accessible, and hospitable for underrepresented students. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 22-26).

Grigg, S., Perera, H. N., McIlveen, P., & Svetleff, Z. (2018). Relations among math self efficacy, interest, intentions, and achievement: A social cognitive perspective. *Contemporary Educational Psychology, 53*, 73-86.

Hadwin, A., & Oshige, M. (2011). Self-regulation, coregulation, and socially shared regulation: Exploring perspectives of social in self-regulated learning theory. *Teachers College Record, 113*(2), 240-264.

Hadwin, A. F., Wozney, L., & Pontin, O. (2005). Scaffolding the appropriation of self-regulatory activity: A socio-cultural analysis of changes in teacher–student discourse about a graduate research portfolio. *Instructional Science, 33*(5–6), 413–450.

Huang, X. (2017). Example-based learning: Effects of different types of examples on student performance, cognitive load and self-efficacy in a statistical learning task. *Interactive Learning Environments, 25*(3), 283-294.

Hushman, C. J., & Marley, S. C. (2015). Guided instruction improves elementary student learning and self-efficacy in science. *The Journal of Educational Research, 108*(5), 371-381.

Janssen, J., Erkens, G., Kirschner, P. A., & Kanselaar, G. (2012). Task-related and social regulation during online collaborative learning. *Metacognition and Learning, 7*(1), 25–43.

Järvelä, S., & Hadwin, A. F. (2013). New frontiers: Regulating learning in CSCL. *Educational Psychologist, 48*(1), 25-39.

Jeong, H., & Hmelo-Silver, C. E. (2016). Seven affordances of computer-supported collaborative learning: How to support collaborative learning? How can technologies help?. *Educational Psychologist, 51*(2), 247-265.

Jepson, A. and Perl, T. (2002). Priming the pipeline. *SIGCSE Bulletin, 34*, 2, 36-39.

Johnstone, B. (2017). *Discourse analysis*. John Wiley & Sons.

Kallia, M., & Sentance, S. (2018, October). Are boys more confident than girls? The role of calibration and students' self-efficacy in programming tasks and computer science. In *Proceedings of the 13th Workshop in Primary and Secondary Computing Education* (pp. 1-4).

Kong, S. C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education*, *127*, 178-189.

Kumpulainen, K., & Mutanen, M. (1999). The situated dynamics of peer group interaction: An introduction to an analytic framework. *Learning and Instruction, 9*(5), 449-473.

Lajoie, S. P., & Lu, J. (2012). Supporting collaboration with technology: Does shared cognition lead to co-regulation in medicine? *Metacognition and Learning*, *7*(1), 45–62.

Lee, W., Lee, M. J., & Bong, M. (2014). Testing interest and self-efficacy as predictors of academic self-regulation and achievement. *Contemporary Educational Psychology*, *39*(2), 86-99.

Lishinski, A., Yadav, A., Good, J., & Enbody, R. (2016, August). Learning to program: Gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance. In *Proceedings of the 2016 ACM Conference on International Computing Education Research* (pp. 211-220).

Margolis, J., Estrella, R., Goode, J., Holme, J. J., & Nao, K. (2017). *Stuck in the shallow end: Education, race, and computing*. MIT press.

Mason, S. L., & Rich, P. J. (2020). Development and analysis of the Elementary Student Coding Attitudes Survey. *Computers & Education*, 103898.

Master, A., Cheryan, S., Moscatelli, A., & Meltzoff, A. N. (2017). Programming experience promotes higher STEM motivation among first-grade girls. *Journal of Experimental Child Psychology*, *160*, 92-106.

Mazman, S. G., & Altun, A. (2013). The effect of introductory to programming course on programming self-efficacy of CEIT students. *Journal of Instructional Technologies & Teacher Education*, *2*(3), 24-29.

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM*, *49*(8), 90-95.

McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochemia medica*, *22*(3), 276-282.

Mejias, M., Jean-Pierre, K., Washington, G., & Burge, L. (2019, February). Underrepresented Groups Threats to Belonging in Computing. In *2019 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)* (pp. 1-4). IEEE.

Mercer, N., Wegerif, R., & Dawes, L. (1999). Children’s talk and the development of reasoning in the classroom. *British Educational Research Journal*, *25*(1), 95–111.

Missiroli, M., Russo, D., & Ciancarini, P. (2016). Learning Agile software development in high school: An investigation. *In Proceedings of the 38th International Conference on Software Engineering Companion - ICSE ’16* (pp. 293–302). Austin, Texas: ACM Press.

Mitchell, A., Purchase, H. C., & Hamer, J. (2009, July). Computing science: what do pupils think?. In *Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education* (pp. 353-353).

Mohamed, A. (2019, May). Designing a cs1 programming course for a mixed-ability class. In *Proceedings of the Western Canadian Conference on Computing Education* (pp. 1-6).

Moyer, L., Klopfer, M., & Ernst, J. V. (2018). Bridging the arts and computer science: Engaging at-risk students through the integration of music. *Technology and Engineering Teacher*, *77*(6), 8-12.

Ogan, C., Herring, S. C., Robinson, J. C., & Ahuja, M. (2009). The more things change, the more they stay the same: Gender differences in attitudes and experiences related to computing among students in computer science and applied information technology Programs. Presented at *Annual Meeting of the International Communication Association*.

Pajares, F. (2002). Gender and perceived self-efficacy in self-regulated learning. *Theory into Practice, 41*(2), 116-125.

Panadero, E., Kirschner, P. A., Järvelä, S., Malmberg, J., & Järvenoja, H. (2015). How individual self-regulation affects group regulation and performance: A shared regulation intervention. *Small Group Research, 46*(4), 431–454.

Pietarinen, T., Vauras, M., Laakkonen, E., Kinnunen, R., & Volet, S. (2019). High school students' perceptions of affect and collaboration during virtual science inquiry learning. *Journal of Computer Assisted Learning, 35*(3), 334-348.

Porter, L., Guzdial, M., McDowell, C., & Simon, B. (2013). Success in introductory programming: What works?. *Communications of the ACM, 56*(8), 34-36.

Potter, J. (1998). Discursive social psychology: From attitudes to evaluative practices. *European Review of Social Psychology*, *9*(1), 233-266.

Rachmatullah, A., Akram, B., Boulden, D., Mott, B., Boyer, K., Lester, J., & Wiebe, E. (2020). Development and validation of the middle grades computer science concept inventory (MG-CSCI) assessment. *EURASIA Journal of Mathematics, Science and Technology Education*, *16*(5), em1841.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11).

Roll, I., Aleven, V., McLaren, B. M., & Koedinger, K. R. (2011, June). Metacognitive practice makes perfect: Improving students' self-assessment skills with an intelligent tutoring system. In *International Conference on Artificial Intelligence in Education* (pp. 288-295). Springer, Berlin, Heidelberg.

Ruis, A. R., Rosser, A. A., Quandt-Walle, C., Nathwani, J. N., Shaffer, D. W., & Pugh, C. M. (2018). The hands and head of a surgeon: Modeling operative competency with multimodal epistemic network analysis. *The American Journal of Surgery*, *216*(5), 835-840.

Saab, N., van Joolingen, W., & van Hout-Wolters, B. (2012). Support of the collaborative inquiry learning process: Influence of support on task and team regulation. *Metacognition and Learning*, *7*(1), 7–23.

Sax, L. J., Lehman, K. J., Jacobs, J. A., Kanny, M. A., Lim, G., Monje-Paulson, L., & Zimmerman, H. B. (2017). Anatomy of an enduring gender gap: The evolution of women's participation in computer science. *The Journal of Higher Education, 88*(2), 258-293.

Sevin, R., & DeCamp, W. (2016). From playing to programming: The effect of video game play on confidence with computers and an interest in computer science. *Sociological Research Online, 21*(3), 14-23.

Shaffer, D. W., Collier, W., & Ruis, A. R. (2016). A tutorial on epistemic network analysis: Analyzing the structure of connections in cognitive, social, and interaction data. *Journal of Learning Analytics, 3*(3), 9–45.

Shah, N., Lewis, C. M., & Caires, R. (2014). Analyzing equity in collaborative learning situations: A comparative case study in elementary computer science. In *Proceedings of the 11th International Conference of the Learning Sciences*.

Sheldrake, R. (2016). Differential predictors of under-confidence and over-confidence for mathematics and science students in England. *Learning and Individual Differences, 49*, 305-313.

Siebert-Evenstone, A. L., Irgens, G. A., Collier, W., Swiecki, Z., Ruis, A. R., & Shaffer, D. W. (2017). In Search of Conversational Grain Size: Modelling Semantic Structure Using Moving Stanza Windows. *Journal of Learning Analytics*, *4*(3), 123-139.

Tsan, J., Vandenberg, J., Zakaria, Z., Wiggins, J. B., Webber, A. R., Bradbury, A., ... & Boyer, K. E. (2020, February). A Comparison of Two Pair Programming Configurations for Upper Elementary Students. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 346-352).

Tran, Y. (2018). Computer programming effects in elementary: Perceptions and career aspirations in STEM. *Technology, Knowledge and Learning, 23*(2), 273-299.

Vandenberg, J., Rachmatullah, A., Lynch, C., Boyer, K.E., & Wiebe, E. (2021). The relationship of CS attitudes, perceptions of collaboration, and pair programming strategies on upper elementary students' CS learning. In *26th ACM Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2021)*, June 26–July 1, 2021, Virtual Event, Germany.

Wagner, I. (2016). Gender and performance in computer science. *ACM Transactions on Computing Education (TOCE)*, *16*(3), 1-16.

Wang, H., Wang, C., & Wu, F. (2020, August). How Multimedia Influence Group Interaction in STEM Education An Epistemic Network Analysis for Online Synchronous Collaborative Learning. In *2020 International Symposium on Educational Technology (ISET)* (pp. 303-306). IEEE.

Werner, L. L., Hanks, B., & McDowell, C. (2004). Pair-programming helps female computer science students. *Journal on Educational Resources in Computing (JERIC)*, *4*(1), 4-es.

Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming. *IEEE software*, *17*(4), 19-25.

Yardi, S., & Bruckman, A. (2007, September). What is computing? Bridging the gap between teenagers' perceptions and graduate students' experiences. In *Proceedings of the third international workshop on Computing education research* (pp. 39-50).

CHAPTER 5: CONCLUSION

Introduction

The preceding work details the process of validating an instrument that assesses upper elementary students' attitudes toward computer science using terminology they understand, and then applying the instrument in a learning analytics study as a way of exploring how student computer science attitudes and performance manifest in their talk, collaboration, and learning. Below are brief summaries of prior chapters, the overall contributions this work has and will make, and future research.

Summary

Chapter 2

We utilized the time-intensive and qualitatively rich process of cognitively interviewing approximately 90 upper elementary students over three iterative studies in four diverse contexts to determine how they understood commonly used computer science terminology. The initial study idea arose from a series of conversations around whether modifying the wording of a previously validated instrument, that centered on the psychological concepts of self-efficacy and outcome expectancy, to include new CS terms and concepts would elicit the information we sought from students of this age. We turned to Karabenick et al. (2007) for guidance on how to employ the cognitive interviewing process with rigor and developed a protocol for querying young students on their thinking and understanding of specific instrument items.

After each round of data collection, we thematically analyzed student responses and made determinations regarding the alignment of their understandings with our item wording intentions. These discussions, informed by the thematic data, resulted in several significant changes to the instrument. These changes include dropping an item because students appeared to interpret it as redundant, splitting two items into additional items (in case, one item became two, and in another one item became three), and completely rewriting an item for clarity (see Table 3, chapter 2).

Because students were unable to provide consistent and appropriate definitions for computer science and computer programs, and because we were most interested in computer programming as an activity, we shifted the wording of the items to include the phrase students used most often—code or coding. We found marked differences in students' experiences with and interests in coding, but they were largely able to respond to the items in ways that indicated they understood what was being asked of them. This qualitative validation of the instrument thus concluded, and we set about collecting quantitative data so we could establish the instrument's validity and reliability.

Chapter 3

To establish the instrument's, now called the Elementary Computer Science Attitudes survey (E-CSA), validity and reliability we utilized both classical test theory and item response theory-Rasch all while adhering to the Standards for Educational and Psychological Testing proposed by the American Educational Research Association, American Psychological Association, and National Council on Measurement in Education (AERA, APA, & NCME, 2014).

After having established that a two-dimensional model was both theoretically most appropriate and provided the best fit, we used Differential Item Functioning (DIF) to evaluate if certain groups (ie., gender or race/ethnicity) answer certain items statistically differently. Although one item demonstrated DIF for race/ethnicity (*Knowing how to code computer* *programs will help me in engineering*) we opted to keep it because other metrics indicated it was a quality item and we felt it pointed to voids in practice and policy that could be addressed.

We then ran a confirmatory factor analysis using the previously established two-dimensional model and permitted the residuals of several items to correlate due, in part, to their linked wording and sequence. Reliability was established using both Cronbach's alpha and plausible value for the two constructs of self-efficacy and outcome expectancy. We then turned our attention to exploring the relationship between students' CS attitudes and performance, and we found that self-efficacy had a significant positive correlation with CS conceptual understanding, but outcome expectancy did not.

Lastly, because computer science has long struggled with not being inclusive to females and certain students of color, we wanted to examine the relationship between gender and race/ethnicity, and students' self-efficacy and outcome expectancy. In alignment with CS literature, we found that males had higher CS self-efficacy and outcome expectancy beliefs than females. However, we did not find that there were any differences between students' self-efficacy and outcome expectancy beliefs by race/ethnicity.

Chapter 4

Having established the E-CSA's validity and reliability, we were interested in using the instrument as a measure in a study of upper elementary students' collaborative learning. In particular, we wanted to see how students' collective regulation of learning may have manifested itself in their discourse as they pair programmed. To do this, we utilized a previously established coding framework to code 12 student dyads' transcripts. The framework centered on two aspects: where the students were in the regulation of learning cycle and the intention of their speech (ie., to be collaborative, to disagree, to express confusion, etc).

These coded transcripts were imported into the Epistemic Network Analysis (ENA) tool along with the students' scores on the self-efficacy subscale of the E-CSA and the CS performance measure—both scores bifurcated by a median split into High and Low. Dyads with both members scoring High became High Status dyads, dyads with both members scoring Low became Low Status dyads, and dyads with members scoring a mixture of high and low became Mixed Status dyads. We anticipated the High and Low Status dyads would demonstrate discursive and regulatory behaviors in alignment with established literature. For example, High Status dyads were likely to exhibit highly self-efficacious, well regulated, and symmetrically contributory work, whereas the Low Status dyads would likely express dysfunction, confusion with little effort toward adaptive help-seeking, and a tendency toward off-task behaviors. This is largely what we found. As such, the Mixed Status dyads proved to be the most interesting.

These dyads, using ENA to qualitatively cluster them, fell into three categories and were split by their use of Collaborative, Disagreement, and Confusion coded statements. Cross-case analyses revealed that the five dyads examined in the study (one High, one Low, and three Mixed) developed a shared understanding and sought help in unique ways. Shared understanding was often developed through question asking or reiteration of what one partner said. Help seeking largely occurred within the dyads; the students tended to rely upon one another for information and sought external, adult assistance at certain, pressing times. Exceptions to these statements regarding shared understanding and help seeking include two all-boy dyads. In these cases, the boys were off-task or engaged in highly critical talk with one another.

Contributions

This work relies on several related fields of study, including survey design and analysis, computer science education, regulation of learning, and visual learning analytics. The contributions of the preceding chapters include the following:

- Pioneering the cognitive interviewing process with young students in a computer science context
- Validating the E-CSA instrument using rigorous methods, using both classical test theory (CTT) and item response theory, the latter of which provides more evidence for validity than CTT.
    - Establishing a largely psychometrically bias-free instrument for use with upper elementary students
- Applying the newly validated instrument in a study of young students' collaborative discourse, with a novel analytic approach, with findings that add to the literature on mixed groups (ie., groups or dyads whose members have different self-efficacies and knowledge in the domain).

Future Research

Validation work is truly never complete. As such, it is important to continue to gather both qualitative and quantitative data on students' shifting understanding of computer science and related terminology, and to what extent they express interest in computing. All three studies would benefit from a larger and more diverse sample of students. Similarly, new attention to remote learning and online digital experiences ought to be considered. Regarding the final study, those engaged in human-computer interaction and learning analytics may consider the value of embedding in-system tutors and prompts as well as visual cues to assist students' collaboration.

References

American Educational Research Association, American Psychological Association, National Council on Measurement in Education. (2014). *Standards for educational and psychological testing*. Washington, DC: AERA.

Karabenick, S. A., Woolley, M. E., Friedel, J. M., Ammon, B. V., Blazevski, J., Bonney, C. R., ... & Kelly, K. L. (2007). Cognitive processing of self-report items in educational research: Do they think what we mean?. *Educational Psychologist, 42*(3), 139-151.

APPENDICES

Appendix A

Final E-CSA Instrument

1. I would like to use coding to make something new.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

2. If I learn coding, then I can improve things that people use every day.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

3. I am good at building code.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

4. I am good at fixing code.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

5. I am interested in how code makes computer programs work.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

6. Using code will be important in my future jobs.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

7. I want to use coding to be more creative in my future jobs.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

8. Knowing how to code computer programs will help me in math.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

9. Knowing how to code computer programs will help me in engineering.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

10. Knowing how to code computer programs will help me in science.

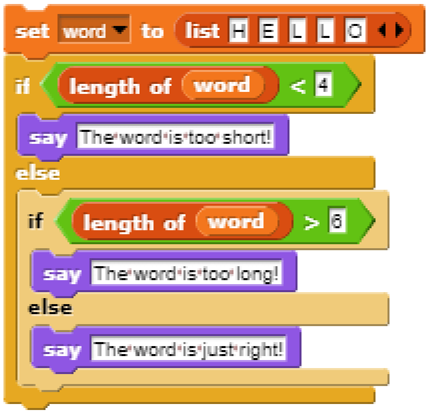
Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

11. I believe I can be successful in coding.

Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree

Appendix B

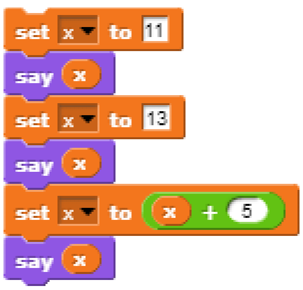
E-CSCA Instrument



What will be said after this code has run?

- The word is too short!
- The word is too long!
- The word is just right!
- Nothing will be said.

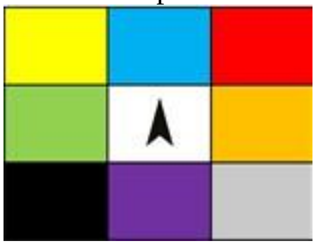
Which of the following can be used to replace  , so that the code will have ***z is equal to 12***?



What is said when this code is run?

- ‘11’ then '13' then '18'
- ‘11’ then '11' then '11'
- ‘x’ then ‘x’ then ‘x+5’
- ‘18’ then ‘18’ then ‘18’
- [Nothing will be said]
- This code will cause an error

Look at the picture below!

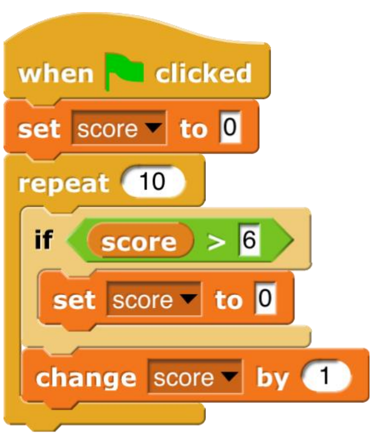


The arrow is heading to the **blue** tile. If you are going to move the arrow to the red tile using the following code, which part of the code needs to be changed?

- ○ Nothing needs to be changed
- ○ Change the block number 3 to 

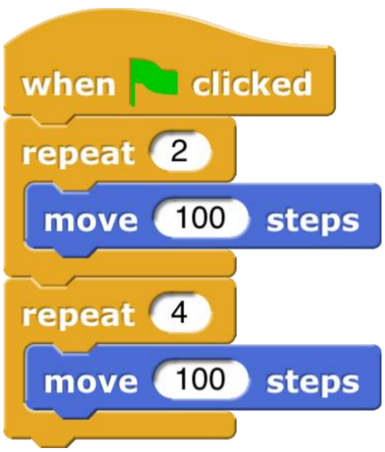
- ○ Change the block number 2 to 

- ○ Move the block 1 to after the block 4



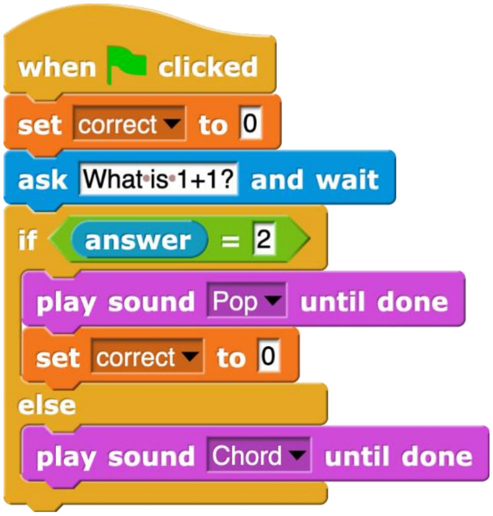
What will the score be after the code runs?

- 0
- 1
- 3
- 10



How many steps will the sprite move after the code runs?

- 100
- 200
- 400
- 600



What will happen if the user enters 3?

- "Pop" plays and correct is 1.
- "Chord" plays and correct is 0.
- "Pop" plays and correct is 0.
- "Chord" plays and correct is 1.

Which of the following should replace 

so that the code will say *“Hello Girls and Boys”?*

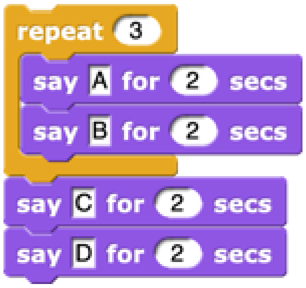
- 

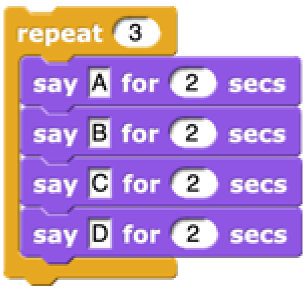
- 

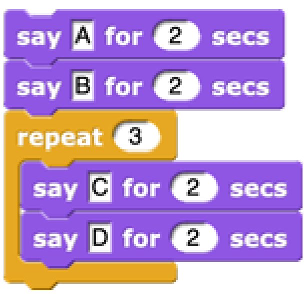
- 

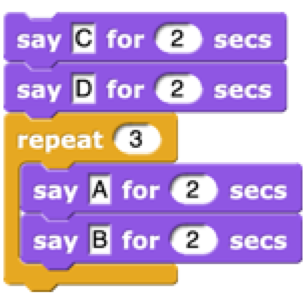
- 

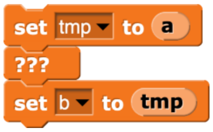
Which lines of code will result in the output saying 'ABABABCD'?





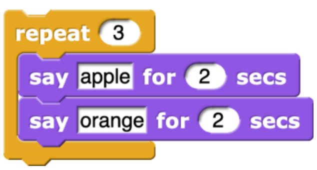






a, b and tmp are variables with values. Which of the following can be used to replace ??? so that the code will switch the values of a and b?

- ○ set tmp to b
- ○ set tmp to tmp
- ○ set a to b
- ○ set b to a



What will be said when this code is run?

○ apple
apple
apple
orange
orange
orange

○ apple
orange
apple
orange
apple
orange

- Nothing will be said
- It will be different each time you run it

If you want to write code that asks a user to type in a sentence, then reports back to the user the number of times the letter ‘e’ appears in that sentence, which of these things would your blocks **<u>NOT</u>** need to be able to do:

- Compare two letters to each other to determine if they are the same
- Display text on the screen
- Convert letters into numbers and numbers into letters
- Store user entered information

The following code is supposed to say “15.”

What needs to be changed in this code for this to happen?

- Change the block number 3 to 

- Change the block number 2 to 

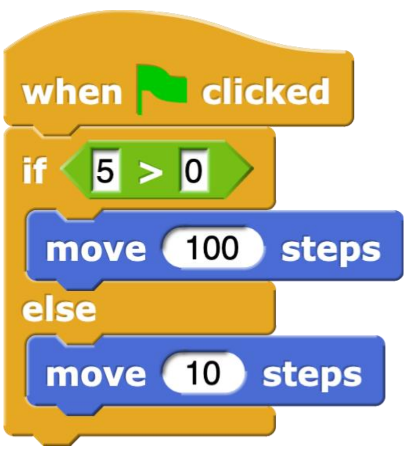
- Change the block number 2 to 

- Nothing needs to be changed

The following code needs to say ‘strawberry’ **six** times. What changes need to be made, if any, for this to happen?

- Nothing, the sprite will say ‘strawberry’ six times
- Block number 2 should come out of the repeat block
- Another  should be added to the repeat block

- The block number 3 should go inside the repeat block



How many steps will the sprite move?

- 5
- 10
- 100
- 110

Which of these answers runs the same blocks in the same order?


when clicked
repeat 4
move 100 steps
wait 1 secs
turn 90 degrees


when clicked
wait 1 secs
repeat 4
move 100 steps
turn 90 degrees


when clicked
repeat 3
move 100 steps
turn 90 degrees
wait 1 secs
move 100 steps
turn 90 degrees


when clicked
repeat 4
move 100 steps
turn 90 degrees
wait 1 secs

Put these mixed up instructions for going to recess as your teacher would like using **THREE** steps.

Items	Click to write Group 1
Run wildly down the hall	
Line up	
Push in my chair	
Yell "It's recess!"	
Walk with class to recess	
Bounce basketball in classroom	

Clap

Clap

Cheer

The above instructions can be written using loops as:

(Clap) x2

Cheer

Which of these answers show the instructions rewritten using loops?

Do three pull-ups.

Do three pull-ups.

Drink water.

Do three pull-ups.

Do three pull-ups.

Drink water.

Do three pull-ups.

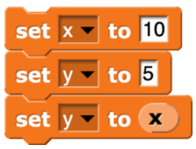
Do three pull-ups.

Drink water.

Have a snack.

- (Do three pull-ups. Do three pull-ups. Drink water.) x4 Have a snack.
- ((Do three pull-ups.) x2 Drink water.) x3 Have a snack.

(Do three pull-ups. Do three pull-ups. Drink water. Have a snack.) x3
((Do three pull-ups. Do three pull-ups.) x2 Drink water.) x3 Have a snack.



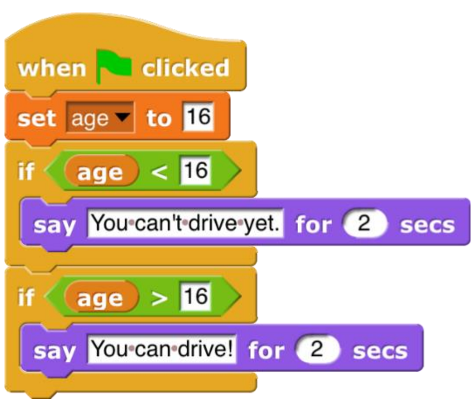
What are the values of **x** and **y** after the above code runs?

x is equal to 10; y is equal to 5

x is equal to 5; y is equal to 5

x is equal to 10; y is equal to 10

x is equal to 5; y is equal to x



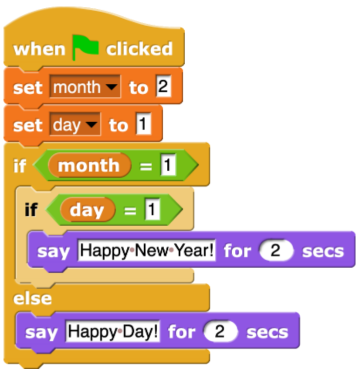
What happens after the code is run?

The sprite says "You can't drive yet."

The sprite says "You can drive!"

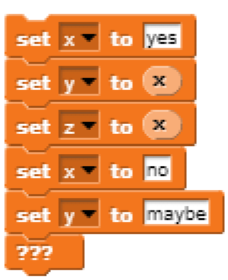
Both of the above.

None of the above.



What will the sprite say after the code is run?

- The sprite says "Happy New Year!."
- The sprite says "Happy Day!"
- The sprite says "February."
- The sprite says "January."



Which of the following can be used to replace  , so that the code will have **z equal to no**?

- 

- 

- 

- does not need an additional block

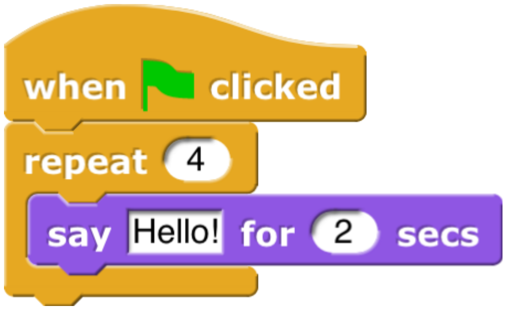
What does this code do?

- Makes sure the value of x is not equal to 10
- Makes sure the value of x is less than 5
- Makes sure the value of x is between 10 and 5
- It always sets x equal to 5
- This code will cause an error

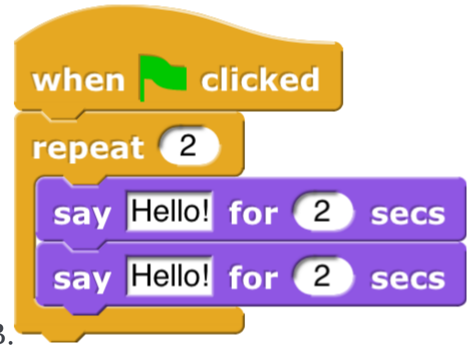
A robot is going to deliver a package to an owner. Below are the steps the robot needs to take to deliver the package.
1. Locate the owner of the package
2. Follow the fastest path from the robot location to the owner’s location
3. Find the fastest path from the robot location to the owner’s location
4. Drop the package
However, there might be small mistake in the order of the steps. Can you find the mistake?

- The order of the steps is just right
- Step number 2 should be after step number 3
- Step number 2 should be after step number 4
- Step number one should be after step numbers 2 and 3

What will the sprite say after code A runs? What will the sprite say after code B runs?



A.



B.

Sprite A:
Hello!
Hello!
Hello!
Hello!

Sprite B:

Hello!
Hello!

Sprite A:

Hello!
Hello!

Sprite B:
Hello!
Hello!
Hello!
Hello!

Sprite A:
Hello! Hello!

Sprite B:

Hello! Hello!

Sprite A:
Hello!
Hello!
Hello!
Hello!

Sprite B:
Hello!
Hello!
Hello!
Hello!