Why Access isn’t Enough: An Analysis of Elementary-Age Students’ Computational Thinking Performance through an Equity Lens

Jean Salac and Diana Franklin

ARTICLE HISTORY
Compiled October 29, 2019

ABSTRACT
With the rise of pre-university Computer Science (CS) and Computational Thinking (CT) instruction worldwide, it is crucial that such instruction is effective for a wide range of learners. While great strides have been made in giving access to students through large-scale school district-level efforts, inequities in learning outcomes may result from tools, curricula, and/or teaching that are appropriate for only a subset of the population. In this paper, we present an analysis of learning outcomes from a large school district’s elementary CT curriculum through an equity lens—one of the few studies with this young age group in a formal school setting. While many students were exposed to CS/CT from an implementation of this scale, we found troubling differences across school performance levels. Our analysis also revealed that reading comprehension and math proficiency were predictive of performance in most CT concepts to varying degrees, and that there were wide gaps between students who were performing below grade level and those who were performing at or above grade level in those subjects. These disparities point to the need for curricular changes and learning strategies to better support students who struggle with reading and math. More insidiously, we found gender and under-represented minority identity to be the most predictive of performance in a small subset of assessment questions. Our results underscore the need for improvement in the way computing is taught to achieve the equity desired from the wider spread of CS/CT instruction.

KEYWORDS
computational thinking, elementary education, Scratch, reading comprehension, math proficiency, equity, race, gender

1. Introduction

With the launch of the CS for All initiative in the US, many American school districts, including San Francisco, Chicago, and New York City, are integrating Computer Science and Computational Thinking instruction at the pre-university level (CS for ALL, n.d.). This trend is not unique to the United States; countries such as New Zealand, Israel, and India have implemented similar programs (Hubwieser et al., 2015). With the growing spread of primary and secondary CS and CT instruction worldwide, it is imperative that such instruction is effective for a broad set of learners.

Unfortunately, primary and secondary CS/CT instruction has not historically resulted in equitable learning. Informal learning opportunities are frequently inaccessible and/or unaffordable to disadvantaged students (Bouffard et al., 2006; DiSalvo, Reid, & Roshan, 2014). Efforts to increase access to CS/CT instruction in the formal school
setting, thereby reaching a broader audience, have sometimes resulted in only the most well-resourced schools getting CS/CT instruction (Fancsali, Tigani, Toro Isaza, & Cole, 2018; Margolis, 2010). Even when CS/CT instruction did make it into the classroom, performance in such a curriculum was found to be associated with higher math, science, and literacy scores (Century, Ferris, & Zuo, n.d.). Bringing CS/CT in front of more students, whether in the formal or informal setting, is not enough; we need to focus on performance in order to achieve the equity proclaimed by initiatives like CS for All.

Nonetheless, the factors contributing to CS/CT performance may also have implicit disparities, such as math proficiency. In studies with participants ranging from ages 10 to university-age, scores on standardized math tests were found to be correlated with performance in a CS/CT course (Byrne & Lyons, 2001; Grover, Pea, & Cooper, 2016; Lewis & Shah, 2012; Wilson & Shrock, 2001). Achievement gaps in math exist for students of color, students who are English language learners (ELL), and students who live in poverty. Among all students in the US, only 20% performed at a below basic level in math. For White students, just 12% met below basic. In contrast, 37% of Black students, 29% of Hispanic/Latinx students, 31% of Native American students, 47% of English Language Learners, and 25% of students who qualified for the National School Lunch Program1 achieved scores that fell below basic levels of performance (Hanushek, Peterson, Talpey, & Woessmann, 2019; NAEP Nations Report Card, n.d.). By 4th grade (ages 9-10), math proficiency gaps are fairly well entrenched and are unlikely to change (NAEP Nations Report Card, n.d.).

In this paper, we examine 4th grade (ages 9-10) learning outcomes from an introductory computational thinking curriculum across school performance, reading comprehension, math proficiency, race/ethnicity, and gender. School performance has been shown to be a proxy for the race, income, and parental involvement of their students (Holme, 2002; Reardon, 2013) and related to resources and teacher turnover rates (Buckley, Schneider, & Shang, 2005). Together with the lack of women and underrepresented minorities in computing, disparities in reading comprehension and math proficiency are also fairly well documented (Doerschuk, Liu, & Mann, 2009; NAEP Nations Report Card, n.d.).

We pursue the following research questions in this study:

- How does school performance influence the learning of the introductory CT topics—events, sequence, & loops?
- How do per-student factors (reading, math, gender, race/ethnicity) predict the learning of CT topics? How do their relative effects change based on the learning goal within each topic?

Our contributions include:

- identifying staggering performance differences across school performance levels,
- determining how different student-level factors can be predictive of performance in specific CT learning goals, thus providing both a bird’s-eye and on-the-ground view of equity in this school district, and
- being one of the few studies to analyze the equity of CT learning outcomes in a formal elementary/primary school setting.

The rest of the paper is structured as follows. In the next section, we present relevant literature on equity in CS education and factors found to be associated with

---

1The National School Lunch Program is an assisted meal program in American schools. It offers free and reduced lunches to children in poverty.
performance in a CS/CT curriculum. In section 3, we describe the Neo-Piagetian theories of cognitive development and models of program comprehension that ground our research, followed by our methodology in section 4. We delineate our results in section 5, by first presenting the larger picture through a comparison across schools and then focusing on student-level factors, both academic and non-academic. Finally, we conclude with the implications of this study and potential future work in section 7.

2. Related Work

We present two bodies of work that this study builds upon—equity in CS education and factors contributing to success in learning CS. Our study extends prior work in the following ways. First, we identify performance differences across school-level and student-level factors in a formal school setting, thus providing a more holistic view of the (in)equities at the elementary school level. Second, we break down our results by specific learning goals and skills to provide a more nuanced picture of CT learning.

2.1. Equity in CS Education

The catalyst for research into the equity of CS education might be the seminal work by Margolis (2010), *Stuck in the Shallow End*, where she found an insidious “virtual segregation” that maintains inequality. She identified structural barriers and curriculum and professional development shortcomings in the integration of CS education in Los Angeles. By tracing the relationships between school structures (e.g. course offerings and student-to-counselor ratios) and belief systems (i.e. teachers’ assumptions about their students and students’ assumptions about themselves), Margolis posits that the race gap in CS exemplifies the way students of color are denied a wide range of occupational and educational opportunities. Her research spawned numerous studies into the obstacles to equitable CS education, both in formal and informal settings.

2.1.1. Opportunities in Informal Settings

Informal CS education, such as after-school programs and summer camps, can be inaccessible and unaffordable to disadvantaged students. In an analysis of two nationally representative datasets of participation in out-of-school activities, Bouffard et al. (2006) found that disadvantaged youth (youth from families with lower incomes and less education) were less likely to participate in out-of-school activities than their peers. If they participated at all, they were participating in fewer activities compared to their peers who came from wealthier and more educated families. They also found that Black and Hispanic youth participated with less intensity in some out-of-school activities, including lessons.

Furthering the connection between out-of-school participation and parental education, DiSalvo et al. (2014) found that the search terms commonly used by parents to find such out-of-school opportunities yielded poor quality results, because they did not have the privilege of education and technical experience when searching for learning opportunities for the children. Among the 840 results from 42 computer related searches, the most powerful and free informal learning tools, such as Scratch and Alice, and free classes and tutorials, such as Khan Academy and Udacity, were completely absent. Instead, the results were filled with summer camps and fee based distance-learning programs. These activities may be prohibitively expensive for students from
2.1.2. Access in Formal Settings

Inequities persist into formal education, as well. In a US-wide survey by Google and Gallup (2016b), a majority of parents (84%), teachers (71%), principals (66%), and superintendents (65%) say that offering CS is just as important as, if not more important than, required courses like math and science. However, only 40% of principals report having at least one CS class where students can learn programming or coding. In addition, male students were more likely than female students to be told by a teacher that they would be good at CS (39% vs 26%). Likewise, Black students (47%) were less likely than White students (58%) to have classes dedicated to CS at the school they attend (Google & Gallup, 2016a).

Studies at the school district level have found similar trends of inequity. In a study of CS course offerings in New York City public schools, Fancsali et al. (2018) revealed that schools that did have CS courses served fewer Black and Latinx students and more White and Asian students. The schools offering CS also served fewer students in poverty and fewer students receiving special education services, and had higher average academic performance and graduation rates. Fancsali et al. (2018) also identified a lack of funding, a lack of staffing, and competing academic priorities as barriers to offering CS or implementing it well. Additionally, in a preliminary study from the American state of Florida, Century et al. (n.d.) found an association between completing more Code.org courses and higher literacy, math and science scores.

2.2. Factors contributing to CS Learning Success

Pea and Kurland (1983) proposed the following cognitive prerequisites to programming from existing literature at the time: (a) math ability, (b) memory capacity, (c) analogical reasoning skills, (d) conditional reading skills, and (e) procedural thinking skills. Since then, there have been many studies analyzing the factors that contribute to success in a CS curriculum, most of which have been at the college level.

At the university level, several studies have cited performance in other subjects as a factor leading to CS success. A study of first-year programming courses by Byrne and Lyons (2001) revealed that scores on a math and science standardized test were strongly correlated with performance in the course, suggesting that CS may require a structure and approach with which science students have some experience and similar cognitive skills used in math. Wilson and Shrock (2001) also found that the number of semesters of high school math courses were predictive of performance on a midterm in an introductory CS class.

Others have attributed success in introductory courses to various cognitive and metacognitive skills. Goold and Rimmer (2000) found personal learning strategies and problem-solving skills to be important to success. In a separate study (Tolhurst et al., 2006), spatial visualization skills were also found to be associated with the success of students, suggesting that different navigational strategies may affect the way in which programmers are able to navigate programming code and form a conceptualization of its major features. On the metacognitive side, Bergin, Reilly, and Traynor (2005) discovered that students who perform well in programming used more metacognitive and resource management strategies than lower performing students, accounting for 45% of the variance in programming performance results. Additionally, a multinational study by Cutts et al. (2006) indicate that students who have a strategic/algorithmic
style of articulation carry on to be successful programmers.

Factors related to students’ belief systems and prior experience have also been found to lead to success. Bergin and Reilly (2005) found that programming performance was strongly correlated with intrinsic motivation, self-efficacy for learning and performance, and students’ perception of their understanding. In addition, Wilson and Shrock (2001) revealed comfort level in the course (i.e. willingness to ask and answer questions, anxiety level while working on assignments, etc) and attribution to luck for success/failure to be predictive of course performance. As for prior experience, Bunderson and Christensen (1995) attributed the high rate of female attrition in CS to the lack of previous experience with computers prior to entering the program. Further, virtually all prior experiences were beneficial for females, while only certain prior experiences correlated with success for males (Taylor & Mounfield, 1994). Hagan and Markham (2000) also discovered that students with experience in at least one programming language at the start of an introductory programming course perform significantly better, and that performance increases with the number of languages. Tying students’ belief systems and prior experience together, Ramalingam, LaBelle, and Wiedenbeck (2004) and Wiedenbeck and Kain. (1994) showed that self-efficacy for programming is influenced by prior experience and increases throughout an introductory programming course. Their results also revealed that the student’s mental model of programming influences self-efficacy and that both the mental model and self-efficacy affect course performance.

By comparison, factors leading to success at the primary and secondary level are less explored. Studies that have been done at the middle-school level (ages 12-14), however, have shown that English and math ability, prior computing experience, and extracurricular technology activities contribute to success in CS learning (Grover et al., 2016; Qian & Lehman, 2016). Lewis et al also found that 5th grade (ages 10-11) student performance on Scratch programming quizzes in a summer camp were highly correlated with their scores on a standardized math test (Lewis & Shah, 2012).

3. Theoretical Framework

We draw upon two sets of theories to frame this study: (1) Neo-Piagetian Theories of Cognitive Development, which explain how children learn and how inequitable childhood learning outcomes can perpetuate in CS/CT, and (2) models for program comprehension, which grounds both the types and the depth of understanding demonstrated by students.

3.1. Neo-Piagetian Theories of Cognitive Development

Piaget’s theory posited that a child’s cognition developed over time based on biological maturation and interaction with the environment (Piaget, 1976). Piaget’s theory has been especially applicable to education in two important ways. First, it spurred the development of new teaching methods that capitalized on the exploratory activities of children themselves. Second, it strengthened the teaching of certain subjects, such as science and math, by cultivating and consolidating the basic thought structures of scientific and mathematical thinking (Demetriou, Shayer, & Efklides, 2016).

Neo-Piagetian theories preserved the strengths of Piaget’s theory while eliminating its weaknesses (Demetriou et al., 2016). They addressed the following weaknesses of Piaget’s theory: (1) it did not sufficiently explain why development between each of
the stages occurs, (2) it did not adequately account for the fact that some individuals move from stage to stage faster than other individuals, and (3) its proposed universal stages of cognitive development have been empirically disproven. We describe four Neo-Piagetian theories to provide some context to our study: Case, Fischer, Commons, and Halford.

Case (1978) and Fischer (1980) proposed that development is not a straight progression through Piaget’s main stages of development, but instead loops over all the stages, each involving their own executive control structures. However, Fischer (1980) argued that environmental and social factors drive development, not individual factors like Case (1978).

To account for environmental/social factors, we look into how gender and under-represented minority identity predict performance. Both identities can influence the learning opportunities available to students in their environments, which may be inequitably distributed.

To account for individual factors, we investigate how an individual student’s reading comprehension and math proficiency predict performance on CT questions. Additionally, Commons (2008) proposed that developmental changes in more hierarchically complex tasks are attributed to the prior completion of simpler tasks. Simpler prerequisite skills needed to learn computing may include reading comprehension and math proficiency, as highlighted in Section 2.

Furthermore, Halford (1993) argued that people understand problems by mapping its parameters to mental models they already have. Existing mental models will differ between individuals based on their environment and prior experiences, supporting the need to analyze student-level factors.

3.2. Models for Program Comprehension

Not only did we want to study factors predictive of performance in a CT curriculum, we also wanted to study both the types and depth of understanding demonstrated by students. To ground the types of understanding, we introduce work by Storey (2005) and Schulte (2008). As for the depth of understanding, we present both Bloom’s and SOLO taxonomies.

3.2.1. Types of Understanding

Storey (2005) synthesized four models of program comprehension. These models include top-down (Brooks, 1983), bottom-up (Pennington, 1987; Shneiderman & Mayer, 1979), systematic (Littman, Pinto, Letovsky, & Soloway, 1987), and opportunistic comprehension (Letovsky, 1987). She also further contextualized these models by differentiating them based on human characteristics, program characteristics, and the context for various comprehension tasks.

Schulte (2008) extended Storey’s work through the Block model. The Block model introduces a duality between “structure” and “function” across three dimensions and four levels of specificity. Two dimensions fall under “structure”—text surface and program execution (data and control flow)—and function (goals of the program) is its own dimension. The dimensions and levels form a table, where each cell highlights one aspect of the understanding process. The cells are designed to be movable, thus allowing for the development of different learning paths. In the Block model, the ultimate goal is for students to build an abstract and general mental model automatically (i.e. unconsciously, so that cognitive resources are freed). The Block model generalizes
program comprehension models by enabling students to build their own strategies to progress through the different cells.

In this study, we focus on the structural understanding aspect of the Block model, since it is often the goal for students building their own projects, which is the case in the curriculum in this study. Two academic skills—reading comprehension and math proficiency—developed at this age range may be associated with the two dimensions of structural understanding. Reading comprehension may be associated with the text surface dimension while math proficiency may be tied with the program execution dimension.

3.2.2. Depth of Understanding

To better contextualize the depth of understanding demonstrated by students, we draw upon Bloom’s and SOLO taxonomies. Bloom’s arranges learning objectives by cognitive complexity (Bloom, 1956; Conklin, 2005), while SOLO arranges learning outcomes by structural complexity (Biggs & Collis, 1982).

Bloom’s taxonomy is comprised of six levels in increasing order:

1. Remember: Recall basic facts and concepts
2. Understand: Explain ideas or concepts
3. Apply: Use information in new situations
4. Analyze: Draw connections among ideas
5. Evaluate: Justify a stand or decision
6. Create: Produce new or original work

Although students do create their own projects—the highest level of Bloom’s taxonomy, the “low floors, high ceiling” design philosophy of the Scratch programming language can result in students using code they do not truly understand (Brennan & Resnick, 2012). Thus, our focus in this study is the lower two levels of Bloom’s taxonomy—remembering and understanding.

To more deeply classify the understanding demonstrated, the SOLO taxonomy was used. The SOLO taxonomy is comprised of five hierarchical levels of understanding:

1. Prestructural: Nothing is known about the subject or task.
2. Unistructural: One relevant aspect is known.
3. Multistructural: Several relevant independent aspects are known.
4. Relational: Aspects of knowledge are integrated into a structure.
5. Extended Abstract: Knowledge is generalized into a new domain.

The SOLO taxonomy is frequently compared to the saying of “seeing the trees but not the forest”. Students may only see a tree (unistructural), several trees (multistructural), or the forest (relational). It has also been used to ground the understanding demonstrated by novice programmers in other studies, ranging from university-age (Lister, Simon, Thompson, Whalley, & Prasad, 2006; Sheard et al., 2008) to fourth-grade students, the same age range of the students in this study (Seiter, 2015). The assessment questions in this study were designed to elicit different levels of understanding along the SOLO taxonomy, which will be further explained in the next section.
4. Methods

In this section, we first characterize the school district, describing its demographics and their implementation of the CT curriculum. We next outline their CT curriculum, detailing the topics covered and the duration of each module. Finally, we describe our assessment design and data analysis processes.

4.1. School District Context

The participants in our study were 296 4th grade (ages 9-10) students from a large urban school district, distributed between a high-performing school, 2 mid-performing schools, and a low-performing school. School performance levels were designated by the school district based on characteristics of both students (e.g. percentage of minority students, English language learners, students with special needs, students in poverty, etc) and teachers (e.g. years of experiences, turnover rates, etc).

Student gender was split almost evenly between male and female\(^2\). The participant ethnic breakdown was 32.91% Asian, 28.79% Hispanic/Latinx, 9.49% White, 8.29% Pacific Islander, and 6.33% Black. The remaining students did not report.

Over the course of a school year (approximately 9 months), they were taught three modules in a Constructionist-inspired introductory CT curriculum in Scratch. All teachers in the study underwent the same professional development to teach this curriculum. Our study was conducted in the school district’s second year of implementing this curriculum to minimize any logistical inconsistencies.

4.2. Curriculum

A modification of the Creative Computing Curriculum (Brennan, Chung, & Hawson, 2011), their curriculum was comprised of three modules—an introduction to Scratch, events & sequence, and loops (see Table 1). The events & sequence module included a lesson on parallelism. Each module could take up to 5 class periods and consisted of an unplugged activity followed by projects that students had to create from scratch, i.e. without a starting project.

<table>
<thead>
<tr>
<th>Module</th>
<th>Project</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intro to Scratch</td>
<td>Scratch Surprise</td>
<td>Explore different Scratch blocks</td>
</tr>
<tr>
<td>Events &amp; Sequence</td>
<td>10-Block Challenge</td>
<td>Create different combinations of the 10 listed blocks</td>
</tr>
<tr>
<td></td>
<td>About Me</td>
<td>Create an interactive collage about yourself using events</td>
</tr>
<tr>
<td>Loops</td>
<td>Build a Band</td>
<td>Create a project about the music you like using loops</td>
</tr>
</tbody>
</table>

Table 1. Modules in the Curriculum

\(^2\)Students in our study only identified as either male or female. No students identified as either non-binary or gender-nonconforming.
4.3. Assessment Design

Our assessment design was guided by the Evidence-Centered Design Framework (Mislevy & Haertel, 2006). Domain analysis was informed by the K-12 Computer Science Framework (n.d.) and K-8 learning trajectories for elementary computing from Rich, Strickland, Binkowski, Moran, and Franklin (2017). These overarching goals were narrowed in domain modeling to identify specific knowledge and skills desired. We were also informed by the SOLO taxonomy to design questions targeted at different levels of understanding (Biggs & Collis, 1982).

In this study, the artifacts students created are projects that students create on their own, not remixed from the Scratch community. Therefore, students are able to use certain Scratch blocks to create their own project. The knowledge they should possess is structural content related to control flow (sequence, events, and loops) and individual block actions. More specifically, they should know what event causes a script to run, the order in which blocks run, and the result of those actions on the stage.

The assessment questions were designed by a team of CS education researchers and practitioners (see Table 2). For face validity, questions were then reviewed by a larger group of practitioners and reading comprehension experts. To ensure that questions on the same topic were consistent, Cronbach’s alpha (α) was also calculated on student responses for internal reliability between questions on the same topic. A Cronbach’s alpha value of at least .7 is generally considered acceptable, at least .8 is considered good, and at least .9 is considered excellent.

In addition, item difficulty (P) and item discrimination (D) values were calculated for each question. Item difficulty is the proportion of students who answered the question correctly—the higher the item difficulty value, the easier the question is. Item discrimination is a measure of how well a question differentiates among students on the basis of how well they know the material being tested—the higher the item discrimination value is, the better the question is at differentiating students (Crocker & Algina, 1986).

We originally designed 3 questions on events, 5 questions on sequence, 6 questions on loops, and 1 question on parallelism. After inspecting student responses, we excluded 4 questions from our analysis because their formatting resulted in spurious markings that made student responses unclear. We excluded 3 Explain in Plain English (EiPE) questions because some students drew the stage, instead of describing the code in words, which would result in ambiguous analysis. In this paper, we present a question on events, a question on parallelism, 2 questions on sequence (α=.78), as well as 5 questions on loops. One of the loops questions has 3 sub-questions, asking about the code in, before, and after a loop (see Figure 1). Thus, there were a total of 7 items for loops (α=.82). The scoring scheme for all questions analyzed are shown in Table 3.

4.4. Data Analysis

We performed two sets of data analyses. For a bird’s-eye view of equity, we analyze performance across school levels. For a more detailed view, we compare across per-student factors—reading comprehension, math proficiency, gender, and URM status.

4.4.1. Comparison across School Performance

Since there were 2 mid-performing schools in this study, we selected the one with three classrooms taught by the same teacher for a better comparison with the high- and the
Table 2. Summary of Question Details

<table>
<thead>
<tr>
<th>Question</th>
<th>CT Concept</th>
<th>SOLO Level</th>
<th>P</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 Events Starting 1 Script</td>
<td>Events</td>
<td>Unistructural</td>
<td>.55</td>
<td>.38</td>
</tr>
<tr>
<td>Q2 Events Starting Parallel Scripts</td>
<td>Parallelism</td>
<td>Relational</td>
<td>.27</td>
<td>.25</td>
</tr>
<tr>
<td>Q3 Repeat Iteration Count</td>
<td>Loops</td>
<td>Unistructural</td>
<td>.91</td>
<td>.39</td>
</tr>
<tr>
<td>Q4 Loop Unrolling</td>
<td>Loops</td>
<td>Multistructural</td>
<td>.58</td>
<td>.41</td>
</tr>
<tr>
<td>Q5 Repeat Blocks vs Iterations</td>
<td>Loops</td>
<td>Multistructural</td>
<td>.57</td>
<td>.53</td>
</tr>
<tr>
<td>Q6a Code in Loop</td>
<td>Loops</td>
<td>Multistructural</td>
<td>.76</td>
<td>.61</td>
</tr>
<tr>
<td>Q6b Code Before Loop</td>
<td>Sequence &amp; Loops</td>
<td>Relational</td>
<td>.71</td>
<td>.64</td>
</tr>
<tr>
<td>Q6c Code After Loop</td>
<td>Sequence &amp; Loops</td>
<td>Relational</td>
<td>.67</td>
<td>.64</td>
</tr>
<tr>
<td>EC Nested Loops</td>
<td>Loops</td>
<td>Extended Abstract</td>
<td>.27</td>
<td>.43</td>
</tr>
</tbody>
</table>

Figure 1. Script shown for Q6, which asked about code in, before, and after a loop

low-performing schools, for a total of 204 students. With this large sample size, the power of all tests was at least 80%.

This quasi-experimental analysis followed the completely randomized hierarchical CRH-\(pq(A)\) model. The linear model is as follows:

\[
Y_{ijk} = \mu + \alpha_j + \beta_k(j) + \epsilon_{ij(k)}
\]

where:

- \(Y_{ijk}\) is the question score for the \(i^{th}\) student in classroom \(k\) within school \(j\),
- \(\mu\) is the grand mean of the question score,
- \(\alpha_j\) is the effect of school \(j\),
- \(\beta_k(j)\) is the effect of classroom \(k\) within school \(j\),
- and \(\epsilon_{ij(k)}\) is the error effect associated with \(Y_{ijk}\).

The independent variable in this analysis was the school performance level, with classrooms nested within them. Both the school performance level and individual classrooms were fixed factors. The classrooms in our study were of different sizes,
<table>
<thead>
<tr>
<th>Question</th>
<th>Scoring Scheme</th>
<th>Max Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 Events Starting 1 Script</td>
<td>2pts/correct; -1pt/incorrect</td>
<td>4pts</td>
</tr>
<tr>
<td>Q2 Events Starting Parallel Scripts</td>
<td>2pts/correct; -1pt/incorrect</td>
<td>4pts</td>
</tr>
<tr>
<td>Q3 Repeat Iteration Count</td>
<td>1pt/correct</td>
<td>1pt</td>
</tr>
<tr>
<td>Q4 Loop Unrolling</td>
<td>1pt/correct</td>
<td>1pt</td>
</tr>
<tr>
<td>Q5 Repeat Blocks vs Iterations</td>
<td>2pts/correct; -1pt/incorrect</td>
<td>4pts</td>
</tr>
<tr>
<td>Q6a Code in Loop</td>
<td>2pts/correct; -1pt/incorrect</td>
<td>4pts</td>
</tr>
<tr>
<td>Q6b Code Before Loop</td>
<td>2pts/correct; -1pt/incorrect</td>
<td>2pts</td>
</tr>
<tr>
<td>Q6c Code After Loop</td>
<td>2pts/correct; -1pt/incorrect</td>
<td>2pts</td>
</tr>
<tr>
<td>EC Nested Loops</td>
<td>1pt/correct</td>
<td>1pt</td>
</tr>
</tbody>
</table>

Table 3. Question Scoring Scheme

so we randomly sampled classrooms of 18 students (the smallest classroom size in our study) and ran the linear model based on the sampled classrooms. This process was repeated 1000 times, and the average of the linear model outputs over all iterations was calculated; the mean of the outputs was used.

Because there are three schools, our analysis was performed in two steps to find statistical significance. First, an ANOVA F-test was used to find whether there are any statistically-significant differences between schools. Then, a Fisher-Hayter Post Hoc test was performed pairwise on the three pair choices to determine which pairs’ result differences were statistically significant. Both tests provide \( p < 0.05 \) is statistically significant.

The eta squared \( (\eta^2) \) effect size was also calculated. \( \eta^2 \) measures the proportion of the total variance in a dependent variable (DV) that is associated with the membership of different groups defined by an independent variable (IV) (Cohen, 1988). For example, if an IV has a \( \eta^2 \) of .25, that means that 25% of a DV’s variance is associated with that IV.

### 4.4.2. Comparison Across Per-Student Factors

Complementing our analysis across school performance, we also compare four per-student factors: reading proficiency, math proficiency, gender and under-represented minority (URM) status. Multiple regression analysis was used (1) to see if any of these factors were predictive of performance on the different questions, and (2) to compare the effects of the predictive factors. To be included in this analysis, students needed to have provided information on all factors, resulting in a total of 189 students.

Logistic regression was used for questions where the answers followed a binomial distribution, i.e. there was only 1 correct answer (Q3, Q4, Q6b, Q6c, EC). We estimate the model as follows:

\[
\log \left( \frac{\text{score}_s}{1 - \text{score}_s} \right) = \beta_0 + \beta_1 \text{reading}_s + \beta_2 \text{math}_s + \beta_3 \text{gender}_s + \beta_4 \text{URM}_s \quad (2)
\]

Similarly, log-linear regression was used for questions where the answers followed
a Poisson distribution, i.e. there were multiple correct options so the regression was done on the number of correct options chosen (Q1, Q2, Q5, Q6a). We estimate the model as follows:

\[
\log(\text{score}_s) = \beta_0 + \beta_1 \text{reading}_s + \beta_2 \text{math}_s + \beta_3 \text{gender}_s + \beta_4 \text{URM}_s
\]  

(3)

Reading and math proficiency scores were both normalized and dummy variables were assigned for gender (1 for female, 0 for male) and URM status (1 for URM, 0 for non-URM). We initially ran regression models with interaction terms between reading scores, math scores, and URM status, none of which were statistically significant and were therefore dropped from the model.

### 4.4.3. Reading Proficiency Analysis

Out of the 296 participants, 231 of them had Scholastic Reading Inventory (SRI) assessment scores. The SRI assessment measures reading skills and longitudinal progress on the Lexile Framework for Reading (Lennon & Burdick, 2004). The SRI Technical Guide defines lexile score ranges for four proficiency levels; the ranges for 4th-grade are shown in Table 4 (“Scholastic Reading Inventory Technical Guide”, n.d.). To identify inequities across the different proficiency levels, the ANOVA F-test was used.

To account for the imbalance across the different proficiency levels, Type III Sum of Squares was used. If the overall F-test was statistically significant, the Tukey-Kramer Post Hoc test was performed on each pair of reading proficiency levels to determine which pairs’ result differences were statistically significant.

<table>
<thead>
<tr>
<th>Proficiency Level</th>
<th>SRI Lexile Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Below Basic (Sig. Below Grade Level)</td>
<td>&lt; 540</td>
</tr>
<tr>
<td>Basic (Below Grade Level)</td>
<td>540-739</td>
</tr>
<tr>
<td>Proficient (At Grade Level)</td>
<td>740-940</td>
</tr>
<tr>
<td>Advanced (Above Grade Level)</td>
<td>&gt; 940</td>
</tr>
</tbody>
</table>

Table 4. 4th Grade Reading Proficiency Levels

### 4.4.4. Math Proficiency Analysis

Out of the 291 participants, 285 of them had Smarter Balanced Assessment Consortium (SBAC) math scale scores. Designed based on the US Common Core State Standards (Preparing America’s students for success., n.d.), the SBAC math assessment assesses students’ knowledge of important mathematical facts and procedures and their ability to apply that knowledge in the problem-solving (“Content Specifications for the Summative Assessment of the Common Core State Standards for Mathematics”, n.d.). SBAC defines 4 proficient levels based on different score ranges. Table 5 shows the ranges for 4th grade (Reporting Scores, n.d.). To identify inequities across the different proficiency levels, we used the same analysis procedure as the reading score analysis.

12
Table 5. 4th Grade Math Proficiency Levels

<table>
<thead>
<tr>
<th>Proficiency Level</th>
<th>SBAC Math Scale Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Novice (Sig. Below Grade Level)</td>
<td>&lt; 2411</td>
</tr>
<tr>
<td>Developing (Below Grade Level)</td>
<td>2411-2484</td>
</tr>
<tr>
<td>Proficient (At Grade Level)</td>
<td>2485-2548</td>
</tr>
<tr>
<td>Advanced (Above Grade Level)</td>
<td>&gt; 2548</td>
</tr>
</tbody>
</table>

4.4.5. Non-Academic Factors

Gender was fairly evenly split among the students, with 144 students who identify as male and 152 students who identify as female. As for race/ethnicity, 100 of the students in our study identified as Asian, 81 identified as Hispanic/Latinx, 30 identified as White, 25 identified as Pacific Islanders, and 19 identified as Black. 41 students declined to state or provided no information. For our analysis, we categorized students based on whether they identified as a race/ethnicity that was under-represented (Hispanic/Latinx, Pacific Islanders, Black) or well-represented (Asian, White) in computing. To compare across genders and URM status, we used the ANOVA F-test with Type III Sum of Squares to account for the imbalance.

5. Results

We present three sets of results to better understand the influences of student performance in computer science. First, we present results across schools of different academic performance to understand whether current instruction serves the goal of equity. We then explore student-level factors, academic and non-academic, to explore potential sources of inequity.

In each section, we present overall results across assessment questions. We then provide detailed results for questions that illustrate the types of questions on which students performed similarly or differently across that attribute. Finally, we discuss potential causes or implications of the results.

5.1. Inequities Across Schools: The Big Picture

We begin by comparing overall performance across high-, mid-, and low-performing schools. Broadly, students in high-performing schools showed a good understanding of events and loops. 99% of them knew the number of iterations a repeat loop performs (Q3), 70% could see the relationship between the loop and equivalent sequential code (Q4), and more than 80% of them understood the order of blocks in a loop compared to blocks before and after the loop (Q6). Only two concepts, parallelism (Q2) and nested loops (EC) were beyond their grasp.

However, our results revealed that students at mid- and low-performing schools exhibited a much shallower understanding of loops. While most could specify how many times a repeat loop will iterate, fewer than half could identify the unrolled equivalent of a repeat loop and identify both constructs that repeat actions (repeat loop and sequential code). Comparing between school levels, there were statistically-significant differences between the high- and mid-performing schools on questions which asked...
about advanced loop concepts (Q5, Q6, Q7; see Figure 2). Students in the mid- and low-performing schools differed on questions on events, parallelism and advanced questions on loops (Q1, Q2, Q7, EC). Finally, students in the high-performing school outperformed students in the low-performing school on all questions with statistically-significant differences.

In the next two subsections, we present two sets of questions that highlight the staggering performance gaps between the different levels of schools (events and loops).

Figure 2. Overall Comparison Across Schools with Scores Normalized relative to the High-Performing School

5.1.1. Events starting Single & Parallel Scripts

There were two questions on events — Q1 covered events starting a single script (Figure 3), while Q2 covered events starting parallel scripts (Figure 4).

In Q1, students received two points for every correct script circled and lost one for any incorrect script circled, for 0-4 points.

The scripts below belong to a sprite. Circle all the scripts that run when you click the sprite.

![Figure 3. Q1 Events Starting a Single Script](image)

The overall average score on Q1 was 2.49 (Figure 5a). Across all three schools, there was a statistically-significant difference \( F(2, 144) = 7.43, p < 0.001, \eta^2 = 0.0792 \). Between pairs of schools, there were significant differences between the low-performing school and both the high- and mid-performing schools with a Fisher-Hayter Post Hoc \( p < 0.05 \).

To better understand how students answered, student responses are categorized as: (1) NO correct - students who circled none of the correct answers, (2) BOTH correct & wrong - students who circled some correct and some incorrect answers, (3) ONLY correct - students who circled correct (subset/all) but not wrong answers, and (4) ALL correct & NO wrong - students who circled all the correct answers and none of
the incorrect ones. As shown in Table 6, students in the high-performing school circled correct options most frequently and provided the most complete answers, followed by the mid- and low-performing schools. Conversely, students in the low-performing school circled incorrect options (No Correct, Both Correct/Wrong) most frequently and were most likely to miss correct options, followed by the mid- and high-performing school.

<table>
<thead>
<tr>
<th>Sch</th>
<th>Category</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No Correct</td>
<td>Both Correct/Wrong</td>
<td>ONLY Correct</td>
<td>ALL Correct/NO Wrong</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>19.3%</td>
<td>7.9%</td>
<td>13.6%</td>
<td>59.1%</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>24.5%</td>
<td>15.9%</td>
<td>18.4%</td>
<td>41.1%</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>33.3%</td>
<td>16.7%</td>
<td>23.3%</td>
<td>26.7%</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>24.8</td>
<td>13.8%</td>
<td>18.0%</td>
<td>43.4%</td>
<td></td>
</tr>
</tbody>
</table>

**Table 6. Q1 Qualitative Results**

Q2 assessed students’ understanding of events across multiple scripts versus sequential events in one script (Figure 4). Students were asked to circle the true statements from the following:

a) Pico plays the drum 7 times THEN changes costumes 4 times.
b) Giga plays the drum 7 times THEN changes costumes 4 times.
c) Pico plays the drum AND changes costumes at the same time.
d) Giga plays the drum AND changes costumes at the same time.
e) Pico and Giga both play the drum 7 times THEN change costumes 4 times.

![Figure 4. Q2 Events Starting Parallel Scripts](image)

The correct answers were (a) and (d). Students earned 2 points for each correct answer circled and lost 1 point for each incorrect answer circled, for 0-4 points. Most students struggled with Q2, with an overall average score of 1.11 out of 4 points (Figure 5a). When broken down by school, the average scores were 1.31, 1.4 and 0.53 points.
for high-, mid-, and low-performing schools, respectively. Across all three schools, there is a statistically-significant difference ($F(2, 144) = 7.82, p < 0.001, \eta^2 = 0.0845$). Between pairs of schools, there are significant differences between the low-performing school and both the high- and mid-performing schools with a Fisher-Hayter Post Hoc ($p < 0.05$).

64.37%, 70.13%, and 46.55% of students in high-, mid-, and low-performing schools, respectively, correctly identified Pico’s sequential behavior. However, only 41.38%, 36.91%, and 35.79% of students in high-, mid-, and low-performing schools, respectively, circled Giga’s parallel behavior.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

Some very common errors include: 44.82% circled Giga having sequential behavior, 22.07% circled Pico having parallel behavior, and 53.85% circled the last option (both sprites have sequential behavior). Taking Q1 and Q2 in perspective, the higher frequency of answers with sequential behavior suggest that students may not understand parallelism as deeply as sequential execution in Scratch, with students in the high-performing school significantly outperforming students in the mid- and low-performing schools.

5.1.2. Loop Functionality

Q3 and Q4 ask about basic loop functionality in two different ways. In Q3, students were asked how the number of times the loop would iterate, while in Q4, students were asked to correctly identify the unrolled version (Figure 6).

Students performed very well on Q3. Almost all of the students from each school were able to answer correctly, with 98.85%, 88.31%, and 84.48% of the students in the high-, mid-, and low-performing schools, respectively getting the answer correct (Figure 5b). Comparing the differences in the number of students who answered correctly, we found a statistically-significant difference ($F(2, 144) = 5.05, p < 0.01, \eta^2 = 0.0431$) among the schools. A Fisher-Hayter Post Hoc pairwise analysis revealed a significant difference between the high- and low-performing schools ($p < 0.05$).

In contrast, students struggled on Q4, with only 56.44% overall answering it correctly. Within individual schools, 70.11%, 53.05%, and 44.83% of the students in the high-, mid-, and low-performing schools, respectively, answered correctly (Figure 5b). There is a statistically-significant difference among schools for Q2 ($F(2, 144) = 5.25, p < 0.01, \eta^2 = 0.0539$), with only a significant difference between high- and low-
performing schools from a Fisher-Hayter Post Hoc \((p < 0.05)\).

When we put Q3 and Q4 performance in perspective, we see that while students are able to identify how many times a repeat loop is run, many students do not truly understand what that means. This implies a limited understanding of loop functionality, especially in the low-performing school.

5.1.3. School Performance Discussion

These results show that at all schools, many of the students are learning basics of core computer science concepts. However, the overall goal of equity is not yet being achieved. Students from low-performing schools are more likely to display a surface-level understanding of the concepts. This is shown especially with loop iteration count vs loop unrolling, in which students at all schools can answer how many times the loop iterates but have difficulty answering exactly how many times, and in what order, that loop causes them to run.

In order to address such inequity, curricular updates, teaching strategies, or learning strategies could be developed. To do so, however, we need to better understand why some students display so much more understanding than others.

5.2. Per-Student Factors

In order to gain more insight into differences in student performance, we looked at two categories of student factors — academic and non-academic factors. Academic factors included reading and math proficiencies, and non-academic factors included gender and URM \(^3\) status, i.e. whether or not a student identified as a race/ethnicity under-represented in computing. A summary of the regressions is presented in Table 7—the larger the absolute value of the coefficient, the more predictive that factor was of that question score.

---

\(^3\)Under-represented racial/ethnic minorities include Black/African-American, Hispanic/Latinx, Native Hawaiian & Pacific Islanders. There were no students who identified as Native American in our study.
In the following section, we present a deeper dive into the academic factors, comparing the influences of both reading and math proficiencies.

<table>
<thead>
<tr>
<th>Question</th>
<th>Reading</th>
<th>Math</th>
<th>Gender</th>
<th>URM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 Events Starting 1 Script</td>
<td>0.19*</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Q2 Events Starting Parallel Scripts</td>
<td>—</td>
<td>.24*</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Q3 Repeat Iteration Count</td>
<td>0.87*</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Q4 Loop Unrolling</td>
<td>0.57*</td>
<td>—</td>
<td>—</td>
<td>-1.01*</td>
</tr>
<tr>
<td>Q5 Repeat Blocks vs Iterations</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Q6a Code in Loop</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Q6b Code Before Loop</td>
<td>0.75**</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Q6c Code After Loop</td>
<td>0.57*</td>
<td>0.55*</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>EC Nested Loops</td>
<td>—</td>
<td>0.67*</td>
<td>-0.81*</td>
<td>-1.1*</td>
</tr>
</tbody>
</table>

* p < .05, ** p < .01

Table 7. Regressions to Identify Factors Predictive of Question Scores

5.2.1. Academic Factors
We highlight two questions where reading proficiencies were predictive: Q3 Repeat Iteration Count and Q4 Loop Unrolling. For Q3, reading proficiency was the only per-student factor that was predictive of performance, while both reading proficiency and URM status were predictive of performance in Q4 (see Table 7).

Comparing relative grade level performance, the differences between reading proficiency levels on both questions were statistically-significant (Q3: $F(3, 227) = 4.57, p < .01, \eta^2_p = .056$; Q4: $F(3, 227) = 15.39, p < .01, \eta^2_p = .17$). For Q3, there were statistically-significant differences between the advanced group and both the basic and below basic groups, and between the proficient and the below basic group. For Q4, there were statistically-significant differences between all pairs except between the proficient group and both the advanced and basic group.

For math proficiency, we focus on two questions: Q2 Events Starting Parallel Scripts (Figure 4) and EC Nested Loops (Figure 7). Math proficiency was the only per-student factor predictive of performance in in Q2, while other non-academic factors were also predictive of performance in EC (see Table 7).

There were statistically-significant differences between math proficiency levels on both questions (Q2: $F(3, 281) = 12.27, p < .01, \eta^2_p = .12$, EC: $F(3, 281) = 25.29, p < .01, \eta^2_p = .22$). For Q2, there were statistically-significant differences between the novice group and both the proficient and advanced groups. As for EC, there were statistically-significant differences between all groups except for the proficient and advanced groups.

5.2.2. Academic Factors Discussion
Reading proficiency was predictive of performance in 5 questions. All 5 questions asked about basic functionality and emphasized a text surface understanding over a program execution understanding—the two dimensions of structural understanding in
Extra Challenge: How many times will the "crash" sound play?

Figure 7. Extra Challenge Question on Nested Loops

The Block model (see Table 7). To answer Q1 correctly, students must be able to read and comprehend the event both in the question and in the answer scripts. Similarly, to answer Q6b and Q6c correctly, students must understand how to read the top-to-bottom order of Scratch scripts. Our results further support the idea that learning to program depends on reading comprehension at several stages in the learning process. Just as in reading (Pressley, 2002), it is not enough to decode the letters into words; to succeed, the student needs to make meaning of the sequences of words into instructions (like sentences) and the sequences of instructions into functions or programs (like paragraphs).

In contrast, the questions where math proficiency was predictive emphasized a program execution understanding. For Q2, students needed to understand parallel execution, which most students struggled with. This result is not entirely surprising – the difficulties that students face while learning parallelism and a related concept, concurrency, are very well-documented (Bogaerts, 2014, 2017; Kolikant, 2001; Lewandowski, Bouvier, McCartney, Sanders, & Simon, 2007; Rague, 2011). However, while most older students were able to identify concurrent/parallel behavior, students in our study struggled to identify an age-appropriate presentation of parallelism. This merits future work into the mental models younger learners have about parallelism, as well as the skills associated with building appropriate mental models. For the extra challenge
question on nested loops, a understanding of loops may be related to an understand-
ing of multiplication, i.e. multiplication can be conceptualized as repetitive addition. If
students do not have a firm grasp on multiplication, they may struggle to understand
repetitive program execution.

Taking both academic factors into consideration, our results reveal that students
must first be able to read and comprehend (1) the words in the blocks themselves
and (2) the structure of scripts in Scratch before they are able to tackle a higher-
order understanding of program execution, at which math proficiency becomes more
predictive.

In terms of equity across different proficiency levels, our results reveal that the clos-
est proficiency levels performed similarly, except in certain loop questions (Table 8
& 9). The performances of the significantly-below- and below-grade-level groups were
significantly different on the loop unrolling question, and most of the advanced loop
questions. The number of significant performance gaps only grows the further the pro-
ficiency levels are from each other, culminating in significant gaps on all questions
between the significantly-below- and the above-grade-level groups. Significant perfor-
mance gaps on loop questions, even between the closest groups, reinforce the need for
improvement in its instruction.

While reading and math proficiencies were predictive of student performance on
most questions (see Table 7), our results revealed non-academic factors—gender and
URM status—to also be predictive, which will be explored in the following section.

<table>
<thead>
<tr>
<th>Q</th>
<th>Reading Proficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SB*B</td>
</tr>
<tr>
<td>Q1</td>
<td></td>
</tr>
<tr>
<td>Q2</td>
<td>—</td>
</tr>
<tr>
<td>Q3</td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td>*</td>
</tr>
<tr>
<td>Q5</td>
<td>*</td>
</tr>
<tr>
<td>Q6a</td>
<td>*</td>
</tr>
<tr>
<td>Q6b</td>
<td>*</td>
</tr>
<tr>
<td>Q6c</td>
<td></td>
</tr>
<tr>
<td>EC</td>
<td></td>
</tr>
</tbody>
</table>

Table 8. Summary of statistically-significant differences between reading proficiency
levels

5.2.3. Non-Academic Factors

There were only two questions where the non-academic factors were predictive — Q4
Loop Unrolling and EC nested loops.

On Q4 Loop Unrolling, URM status was the most predictive factor. Further com-
paring all students who had information on race/ethnicity, not just those who also
had information on gender, reading and math, we found that students who identified
as a well-represented race/ethnicity outperformed those who identified as an under-represented race/ethnicity ($F(1, 253) = 7.29, p < .01, \eta^2 = .11$; see Figure 9a).

The EC question on nested loops (Figure 7), which was not explicitly covered in the curriculum, was the only question in which both gender and URM status were predictive (see Figure 7). Comparing all students with information on gender and race/ethnicity, regardless of whether or not they had information on reading and math proficiencies, an ANOVA F-test revealed that students who identified as male statistically-significantly outperformed students who identified as female ($F(1, 294) = 5.73, p < .05, \eta^2 = .019$; see Figure 9b). Similarly, students who identified as a member of a well-represented race/ethnicity statistically-significantly outperformed students who identified as a member of an under-represented race/ethnicity ($F(1, 53) = 38.77, p < .01, \eta^2 = .13$; see Figure 9c).

<table>
<thead>
<tr>
<th>Q</th>
<th>Math Proficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SB*B</td>
</tr>
<tr>
<td>Q1</td>
<td></td>
</tr>
<tr>
<td>Q2</td>
<td></td>
</tr>
<tr>
<td>Q3</td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td>*</td>
</tr>
<tr>
<td>Q5</td>
<td>*</td>
</tr>
<tr>
<td>Q6a</td>
<td>*</td>
</tr>
<tr>
<td>Q6b</td>
<td></td>
</tr>
<tr>
<td>Q6c</td>
<td>*</td>
</tr>
<tr>
<td>EC</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 9. Summary of statistically-significant differences between math proficiency levels

![Figure 9. Questions with Predictive Non-Academic Factors](image-url)
5.2.4. Non-Academic Factors Discussion

URM status was the most predictive of performance on two questions, Q4 Loop Unrolling and EC Nested Loops. Further, the interactions between URM status and both reading and math proficiencies were not significant. This suggests that there may be other factors associated with being part of an under-represented group that could be impacting their performance in a computing curriculum. Among others, potential factors include spatial skills, which have been found to be tied to STEM achievement (Margulieux, 2019). Males have been measured as having higher spatial ability, potentially due to the toys given to them in early childhood (Linn & Petersen, 1985). Socioeconomic status may also have an impact, as it is linked to academic skills developed in early childhood (Hanushek et al., 2019). Investigation into other such factors associated with being part of an under-represented race/ethnic group and how they influence performance in a computing curriculum merits further study.

We can gain insight into potential factors involved in URM performance by focusing on the question where URM status was most predictive—the extra challenge question on nested loops. This question is unique because it is the only question that (1) covered material which was clearly beyond the curriculum and (2) can be predicted by gender and URM status. Prior research would hypothesize that this could be due to informal computing experience that well-represented students may have, a variable we did not control for (DiSalvo et al., 2014; Jacobs & Bleeker, 2004; Wang, Hong, Ravitz, & Ivory, 2015). Under-represented students may face barriers to accessing informal opportunities such as cost, transportation, and the lack of parental technical familiarity (DiSalvo et al., 2014; Ericson & McKlin, 2012). This further supports the need for in-school, formal computing instruction from a young age, before the disparities between well-represented and under-represented students grow too large.

6. Limitations

There are several limitations to this study. First, the demographics of the school district in the study may not hold for other school systems worldwide. Similar studies in other school districts with different contexts would be necessary for a wider understanding.

Additionally, due to the testing schedule of the school district and the timing of this study (2017-18 school year), the most updated SBAC math scale scores and SRI lexile scores used in this analysis came from different times. The SBAC scores came from Spring 2017, while the SRI scores can come anytime from January to June 2018 because schools have flexibility on when to administer the SRI assessment. As a result, these scores were the closest approximation of their math and reading proficiency at the time they took the CS assessment, not exact measures.

Although we have delineated the associations of various academic and non-academic factors on CT learning outcomes, our study is far from a complete picture. There are other student populations that struggle to read and do math proficiently, such as students with disabilities, which may adversely affect their performance in a CT curriculum. Further, as outlined in section 2, there is a multitude of factors that have been linked to performance in various CS courses at the university-level, which are worth exploring at the elementary and secondary level.
This paper presents an investigation into the factors associated with learning outcomes from a school district’s implementation of an introductory CT curriculum for elementary-age students. Our analysis revealed worrisome differences across school performance levels, with students in the high-performing school significantly outperforming those in the low-performing school in all CT questions. School performance has been associated with the race, income, and parental involvement of their students, as well as resources and teacher turnover rates (Buckley et al., 2005; Holme, 2002). Thus, the relatively poor learning outcomes from students in the mid- and low-performing schools indicates that disadvantaged students, the very students that such a wide implementation is aiming to reach, are getting the short end of the stick.

A deeper dive into per-student factors reveal how academic and non-academic factors can put some students at a disadvantage. Analysis at the student-level revealed reading comprehension to be predictive of most questions that emphasized a text surface understanding, while math proficiency was predictive of questions which emphasized a more program execution understanding. Comparing relative grade-level performance, we found stark performance differences between students who perform below grade-level and those who perform at or above grade-level. Students who are not becoming proficient in reading and math are disproportionately students of color, ELLs, and students who live in poverty—students from communities traditionally marginalized from computing (Hanushek et al., 2019; NAEP Nations Report Card, n.d.). If the associations between reading and math proficiencies and CS/CT learning remain unaddressed, efforts to increase access to formal computing experiences will fall short of addressing issues of equity.

In contrast, identifying as male and/or a member of a race/ethnicity well-represented in computing was predictive of performance in a question not explicitly taught in the curriculum. This indicates that there are still other factors that advantages such students, or conversely, factors that disadvantage students have identities under-represented in computing. Investigation into identifying these other factors merit further study.

These results have broad implications for researchers, instructors, curriculum developers, policymakers and other stakeholders in the movement to integrate CS/CT instruction at the pre-university level. Simply providing CS/CT instruction is not sufficient. Future work can focus on improving the quality of instruction and instructional materials. This includes student-oriented efforts like meta-cognitive learning strategies, debugging strategies, and culturally-relevant curriculum design. There is also work to be done in teacher professional development, recognizing the heavy lift of introducing a new area for classroom teachers. How can we make professional development opportunities and curricular materials clear and educative for the novice CS/CT teacher? This study has shown that it is not enough to provide equitable access to CS/CT instruction; we must also work towards equitable outcomes if we are to truly include students that have been historically marginalized in computing.

Funding
This material is based upon work supported by the National Science Foundation Grant No. 1660871 and Graduate Research Fellowship Program under Grant No. DGE-1746045. Any opinions, findings, and conclusions or recommendations expressed in
this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References


Content specifications for the summative assessment of the common core state standards for mathematics [Computer software manual]. (n.d.).

Crocker, L., & Algina, J. (1986). Introduction to classical and modern test theory. ERIC.


