An Analysis of Middle-School Student Behavior using the TIPP&SEE Metacognitive Learning Strategy

ANONYMOUS AUTHOR(S)

As computer science (CS) instruction matures at the elementary and middle-school levels, new pedagogical approaches, learning strategies, and teaching strategies are being developed. Because of the increase in development between 4th grade (ages 9-10) and 7th grade (ages 12-13), students may respond differently to the same approach. In this paper, we explore middle-grade students' (age 9-13, grades 4-7) use of TIPP&SEE in an intermediate, Scratch-based, Use→Modify→Create programming curriculum: Blinded Curriculum, which was designed to be an intermediate programming curriculum. We present analysis of 307 students' work, investigating student participation and accuracy in the different TIPP&SEE phases. Our findings show that middle-grade students who use TIPP&SEE show similar engagement and answer questions with the similar correctness across a variety of factors such as previous experience with programming (parent-reported). However, among the TIPP&SEE phases, the students demonstrated higher engagement and the most success with answering questions that involve observing an example program operation before ("Observe") making changes to the program. The students demonstrated higher engagement but less accuracy answering questions after making changes to the program ("Explore"). Finally, our results show that student engagement and correctness for a given question is impacted by its placement on the worksheet. These findings reveal important design aspects of TIPP&SEE and how it supports students across a variety of factors, and how students struggle with making "Predictions" and interpreting certain concepts correctly.

CCS Concepts: • **Social and professional topics** → **Computer science education**;

Additional Key Words and Phrases: Computational Science Education; K-12 education; Scratch

ACM Reference Format:

An Analysis of Middle-School Student Behavior using the TIPP&SEE Metacognitive Learning Strategy. In *Proceedings of the 2021 International Computing Education Research Conference (ICER'21), August 10–12, 2021, Virtual Event, Toronto, Canada.* ACM, New York, NY, USA, 11 pages.

1 INTRODUCTION

Equitable and accessible instruction in computer science (CS) and computing is essential for preparing all students to succeed in an increasingly computationally-driven world. It is important that educational opportunities are not only available, but that students develop meaningful and actionable computing knowledge as part of this instruction. While much emphasis has been placed on the creation of introductory CS educational resources and computing courses in high school and higher education, there has been a gap in resources and computing courses for younger students. A study of curricula available to elementary and middle-school students in 2 regions of the United States showed a lack of specification of general purpose programming, object oriented programming, and visual programming [11]. In response to these and other gaps in instructional materials, Blinded Curriculum was designed to focus on elementary and middle-school students [1]. Blinded Curriculum employs a number of pedagogical innovations, including TIPP&SEE

Manuscript submitted to ACM

which has been shown to promote students writing longer programs, writing more code per requirement, using more required blocks in later assignments, and satisfying more assignment requirements [13]. Overall, TIPP&SEE has been shown to increase elementary-school student acquisition of computational knowledge and create more equitable access points for students [13]. In this paper, we explore the ways in which *middle-grade students* interact with the TIPP&SEE pedagogical strategy and if TIPP&SEE supports their acquisition of computing knowledge. In order to understand the ways in which middle-grade students engage with TIPP&SEE, in this paper, we address the following 3 research questions:

- What level of completion and accuracy do middle-grade (4th-7th) students achieve across the 3 sections of TIPP&SEE?
- What factors influence accuracy and completion rates of individual questions?
- How do these findings compare with prior research on TIPP&SEE with younger (4th grade) students?

This paper is organized as follows. In Section 2, we discuss the major theoretical and practical influences that inform this work. In Section 3, we describe the TIPP&SEE strategy that scaffolds student development through the Use→Modify components of the Use→Modify→Create learning framework. Section 4 presents the methodological approach used to investigate the stated research questions. In Section 5, we present the main results of this paper, including overall performance and student-related, classroom-related, and question-related factors that affect student outcomes. In Section 6, we discuss these results relative to related work. The paper concludes with a review of the limitations of our findings and discussion of next steps for this work.

2 THEORY AND PRIOR WORK

In this section, we present the major theoretical and practical influences that shaped this work. This includes prior work on theories concerning learning, teaching strategies, and middle-grade computing education curricula and pedagogy.

2.1 Theoretical Orientation

This work draws on constructionist learning theory and the concepts of the *Zone of Proximal Development* (ZPD) and *instructional scaffolding*. Constructionist learning theory focuses on developing opportunities for learners to construct their own knowledge, rather than the focusing on directly transmitting this knowledge to students through direct instruction [30]. Constructionist activities place students in project-based learning activities so they can learn through connecting what they already know to discoveries they make in the process of creating artifacts of their own [31, 32]. Allowing students the freedom to construct their own understanding can lead to a number of challenges when employed in formal education contexts, such as keeping students on the intended learning trajectory and ensuring they learn specific content. In this process, some students may never reach the conclusions necessary or achieve the learning objectives of the activity. This tension between learner agency and externally defined learning goals, called the *play paradox*, highlights the need for a strategy to support student exploration under sufficient constraints to ensure that specific learning outcomes are achieved [29].

The zone of proximal development describes the link between a student's current abilities and their potential developmental achievements that can be attained through the guidance and encouragement from another, more skilled student [41]. By providing further challenges to some students and through the use of instructional scaffolding— individualized support from the instructor that will be gradually removed— to others, the instructor can leverage each

activity to keep students within their ZPD. This approach both maintains every student's engagement and supports their individualized learning growth [28, 45].

The Use→Modify→Create educational framework combines constructionist learning theory, the ZPD, and instructional scaffolding as a strategy to avoid the play paradox [12, 23]. This approach involves students who are given constraints within which they can explore larger computing concepts, through modifying (within these constraints) a given code base. Use→Modify→Create initiates the learning experience with students in the Use activity, which provides examples of the focal concept being used and involves heavy scaffolding. The Modify activity follows with some of the scaffolds faded away as the students experience a guided exploration to modify the use of the concept. Students can then more directly explore the skills they have learned in a more powerful environment than their present skill level may support, but in a manner that enables them to gain intellectual ownership in this broader context. Finally, the Create activity allows them to explore and play, employing the concepts in ways they invent. This strategy increases students' feeling of ownership over their work and appreciation for the context in which it lies in the broader computing framework [9, 24].

2.2 Pedagogy In middle-grade Computing Education

Early access to computing education promotes equity and learning outcomes for computing education [20, 30, 33]. There is a wide array of computing curricula for elementary and middle-school students, including those based on structured, behaviorist curricula and direct instruction. On the other side of this spectrum are curricula that employ strategies to facilitate constructionist learning. This open-ended, student-centered approach allows students to discover the utility of each coding concept as they can be used in many situations, while gaining confidence in their abilities through feeling intellectual ownership over their own project. [31, 32]. Most computing curricula and programming environments created for middle-grade students are based on constructionist or constructionist-inspired methods. While direct instruction may produce quicker learning outcomes in terms of student familiarity with specific learning objectives, constructionism promotes student engagement and confidence as they encourage students to create programming projects that they are excited about and proud to share with the world. [2, 16] Many of these constructionist learning environments and curricula can trace their roots back to the Epistemology and Learning research group at MIT, including Logo, Scratch, NetLogo, and Lego Mindstorms. [15, 38]. These are also examples of the growing movement of block-based programming environments created to be accessible to students at all levels of prior computing knowledge. The drag-and-drop block-based format functions as scaffolding for students removing the potential for syntax errors and providing visual cues that are more approachable and similar to online games with which students new to programming may have more familiarity. [42–44] Centered in the spectrum of computing curricula is the Use→Modify→Create framework.

2.3 Meta-cognitive Learning Strategies

TIPP&SEE is a metacognitive learning strategy that proceduralizes student engagement to scaffold student learning (while learning from provided code as an example) and improves learning in the Use→Modify step of the Use→Modify→Create pedagogical approach designed for elementary computing instruction. TIPP&SEE is based on reading comprehension strategies [37]. Learning strategies are techniques, principles, and rules that teach students how to learn instead of teaching them specific content. These strategies enable students to learn, solve problems, and complete tasks independently [6]. Meta-cognitive learning strategies in particular enable students to guide themselves through the learning process via a focus on self-regulation and self-motivation [34]. Students who use meta-cognition

have knowledge and awareness about their own thinking, and are able to use this to bolster their learning by applying domain-appropriate learning strategies [7]. Employing meta-cognitive strategies results in students having improved memory, test scores, and lecture comprehension [21, 22].

The motivation behind explicitly teaching meta-cognition is that behaviors of students who naturally use meta-cognitive strategies are intrinsic to the students themselves, and thus non-meta-cognitive learners are further disadvantaged when not taught the process of learning [4]. Thus, to create more equitable learning environments, hidden meta-cognitive behaviors have to be revealed to all learners by teaching meta-cognitive learning strategies. Mnemonic devices are one such meta-cognitive strategy. Mnemonic devices use acronyms to scaffold meta-cognition by prompting lists of information or actions comprising learning strategies [39]. Mnemonic devices are often used along with other meta-cognitive learning strategies for reading comprehension, such as Previewing and Text Structure. Previewing guides students to identify important sections of text and call on related previous knowledge before reading. Text Structure prompts students to recognize the underlying organization of a text and retrieve structure-relevant processing strategies, thus enhancing their comprehension [17]. Teaching such comprehension strategies to middle-grade students in particular has been shown to improve performance on text summarizing, comprehension, memory, and maintenance of performance gains over time [3]. Researchers have thus explored deploying comprehension strategies to create equitable learning opportunities in areas such as mathematics [27], writing [18], history [5], and most recently, CS [15, 37].

Within CS, comprehension strategies are especially crucial in enabling students' comprehension of code throughout a programming assignment, as they are required to read and process text such as individual instructions, sequences of instructions in the form of example code, and their own partially or fully complete code [38]. Indeed, the aforementioned examples of comprehension strategies can aid in this. When previewing, students can identify familiar and new code, connect unfamiliar concepts to prior knowledge, and use that to guess how the new code functions. Using text structures to understand the structure of the specific programming language and environment in which the students are working will allow them to draw on particular strategies for that scenario.

Researchers have shown that successful CS students also employ meta-cognitive comprehension strategies such as self-assessing understanding, creating timelines to track progress, and actively troubleshooting issues [26]. Work on one specific CS comprehension strategy, TIPP&SEE, has demonstrated that upper elementary school-aged students using the strategy have a more sophisticated understanding of CS concepts and increased performance on moderate to hard questions [38]. This strategy is presented in more detail in the following section.

To date, all studies conducted with TIPPSEE have been with 4th grade students. The researchers who introduced TIPP&SEE found that the students who used TIPP&SEE outperformed the control students who used an unmodified Use→Modify→Create approach on almost every moderate and hard question. Prior work also analyzed student artifacts, assessments, and TIPP&SEE worksheets and found that students who use TIPP&SEE are more thorough in their work, write longer scripts, use more blocks taught in the module, and are more accurate in their question responses. Additionally, researchers demonstrated that with the 4th grade students studied, TIPP&SEE is equitable, showing that benefits extend to students who receive free or reduced lunch at school, students with disabilities as identified through district provided demographic data, and students with below-grade-level reading and math proficiency as identified through state testing. Among these students, those who used TIPP&SEE to scaffold a Scratch curriculum with the Use→Modify→Create framework performed similarly to their peers in CS instruction despite typically performing below proficiency on state testing in reading and math. [36]. Finally, prior work demonstrated that TIPP&SEE improves student performance (or at least increases behaviors that promote success) in a wide range of student work, and that

this enhancement extends to all students including students who typically perform below proficiency [13, 36, 37]. In comparison, this current paper focuses on how students engage with the TIPP&SEE worksheet. We investigate student answer behaviors (engagement) and performance, across the different worksheet sections (**TIPP**, **SE**, and **E**xplore) and how their behaviors may be influenced by the students' prior knowledge, the arrangement of the worksheet, their teacher, their grade, and prior experiences with programming and math. Further, while prior work about TIPP&SEE studied 4th grade students, ages 9-10 [13, 36, 37], this paper explores the engagement with TIPP&SEE of middle-grade students between 4th and 7th grade, ages 9 - 13 to understand if there are differences in student engagement between younger elementary-school students and middle-grade students with TIPP&SEE.

3 ENACTING TIPP&SEE

To illustrate how TIPP&SEE was enacted in our curriculum, we first detail Blinded Curriculum, then provide details about TIPP&SEE, and an example of a module from Blinded Curriculum.

3.1 Curriculum

Blinded Curriculum is an intermediate CS curriculum designed for middle-grades learners [1]. The curriculum is designed for use by students with limited previous coding experiences and, after reviewing events and loops, covers intermediate topics including conditional loops, synchronization, and variables. It is designed to provide students with rigorous Scratch [25] coding experiences that bridge the gap between elementary experiences and high school classes. The curriculum is a 2-3 year curriculum with 15 modular units of 3-5 lessons each. In order to increase cultural relevancy to students, lessons are available with youth-relevant themes in 3 strands: multicultural, youth culture, and gaming. Teachers select the thematic strand to use within their classroom. For each module, computing content is the same across strands.

All Blinded Curriculum programs and activities present authentic uses of the content being taught. This means the code that students see as they go through the curriculum reflects the standard approach to solving that problem, regardless of the knowledge level of the problem solver. In this way, the curriculum not only teaches *how* a given concept works but also *when* to use it.

Blinded Curriculum uses 2 learning and pedagogical strategies: Use→Modify→Create [9] and TIPP&SEE [37]. The Use→Modify→Create pedagogical strategy is used to scaffold the conceptual learning. Students first explore and modify a functioning example. Then they apply the new concept(s) to an open-ended create project[1].

3.2 TIPP&SEE

In this section we introduce TIPP&SEE with a focus on how TIPP&SEE is enacted within Blinded Curriculum. TIPP&SEE is a meta-cognitive learning strategy designed to help students learn from provided code. TIPP&SEE is divided into 4 question phases ("Preview", "Observe", "Predict", and "Explore"). These phases were designed to help students explore the example project prior to making modifications.

TIPP stands for:

- **T**itle: What is the title of the project? Does it tell you something about the project?
- **I**nstructions: What do the instructions tell you to do?
- **P**urpose: What is the purpose of this activity?
- **P**lay: Run the project and see what it does! Which sprites are doing the actions?

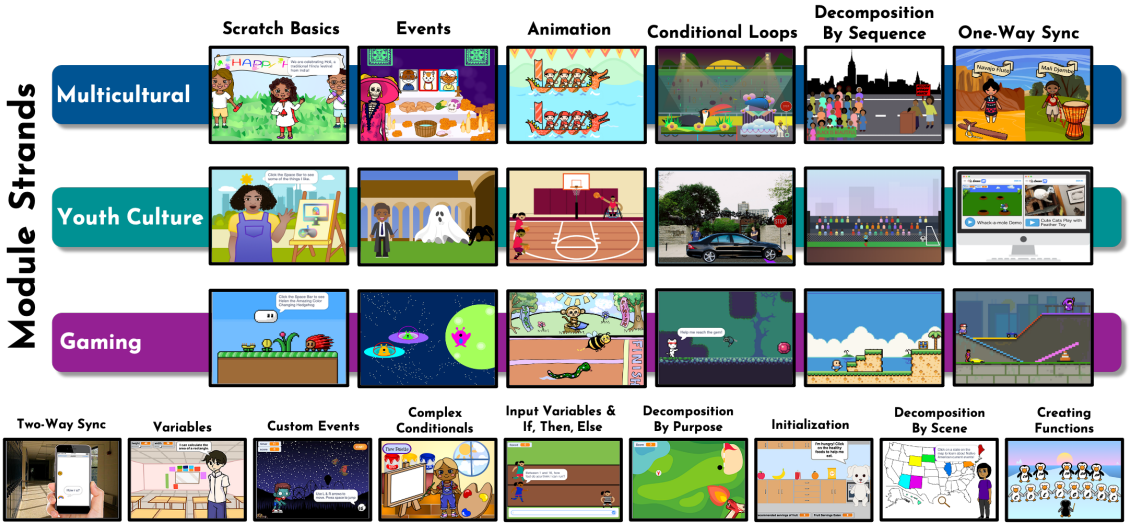


Fig. 1. 3 curricular strands, providing per-module activity choices

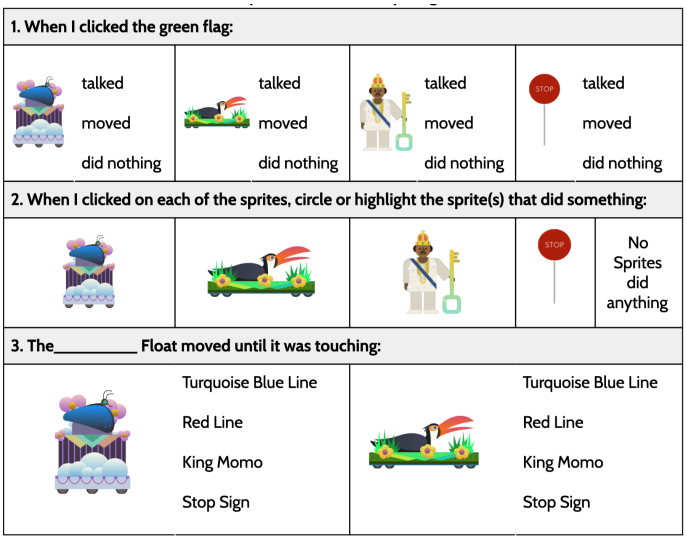


Fig. 2. **TIPP** worksheet guides students through mindfully playing and observing the provided Scratch project.

The **TIPP** portion of TIPP&SEE was developed following scaffolding strategies that were effective in supporting student reading comprehension [10]. Students first "Preview" the project by reading the **T**itle, **I**nstructions, and **P**urpose (**TIP**) of the Scratch project in order to orient the student to both how to interact with the project and the learning goals of the activity. The students then "Observe" what the program does by **P**laying (**P**) the provided program. The questions in the accompanying worksheet are organized to ask the student to identify, for each user *event*, what *sprite* performed what *actions* because Scratch code is organized by these 3 elements.

Figure 2 provides an example of **TIPP** questions asking students about a Scratch project in Blinded Curriculum M4:CondLoops. After an initial discussion about repeated actions and how repeated actions can end after a set number of repetitions or based on a condition becoming true, students are given example code from a project with a Carnival setting reflecting the Multicultural strand. The example program includes scripts for 3 sprites: a Butterfly Float, a Toucan Float, and a king figure. A stop sign sprite is also included in the program, but it has no scripts associated with it. When the green flag is clicked, the Butterfly Float sprite moves until it touches the stop sign sprite, illustrating how a conditional loop works. The questions provided in Figure 2 focus on the final P of **TIPP**, **P**laying, by asking students to interact with the project in particular ways, such as clicking the green flag or clicking certain sprites. Students are then also tasked with noticing resulting movements or sounds from the aforementioned interaction via questions requiring them to distinguish whether a given sprite talked, moved, or did nothing (question 1), note if a given sprite responded to any of the sprites being clicked (question 2), and record when a given sprite stops moving (question 3). These questions are carefully chosen to highlight important aspects of the example code.

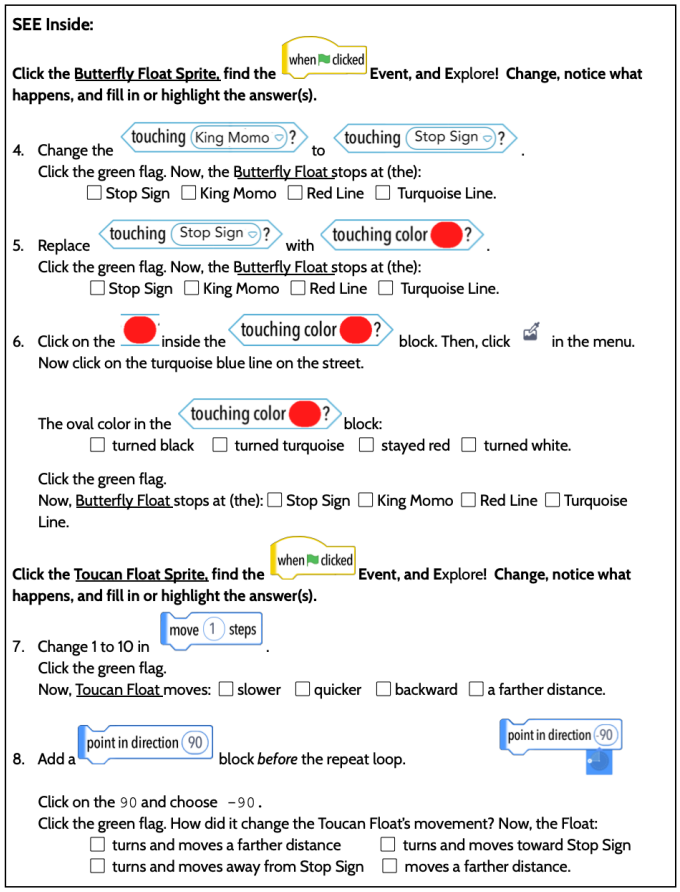


Fig. 3. **SEE** worksheet helps students navigate Scratch and learn blocks through deliberate exploration (tinkering).

SEE stands for:

- **S**prites: Click on the the sprite that you want to learn from or change.
- **E**vents: Look at the `event` blocks starting the scripts.
- **E**xplore: Try different changes to the scripts and observe what happens!

The **SEE** portion of TIPP&SEE refers to that students are then instructed to click the "SEE INSIDE" button on the Scratch interface and answer questions about and with the code. The **SEE** portion of TIPP&SEE was inspired by the work of researchers creating reading comprehension writing structure education plans [8]. The **SEE** portion stands for the following: **SE** is a navigational scaffold to help students find code in Scratch. To find code, students are instructed to click on a specific **S**prite (**S**) and locate a specific **E**vent (**E**). Once they find the code, students are asked to answer a series of "Predict" questions asking them to inspect the code, reflect on what they observed during **P**lay, and then predict what blocks in specific scripts caused the specific actions they observed. Finally, the students **E**xplore (**E**) through making changes to the code and reporting the changes in the behavior of their sprites as a result of their changes. For example, by adding, removing, or moving blocks, or changing the parameters of a specific block, and reporting how it changes the program, the students are able to gain a detailed understanding of that block and how to use it.

Figure 3 provides the **SEE** questions complementing the **TIPP** questions about Conditional Loops introduced earlier in Figure 2. The **SE** navigational scaffold in this example worksheet instructs students to locate the `when green flag clicked` event of the Butterfly Float sprite. Then, students are guided through exploration by way of changing the fields of the `touching` block (question 4), replacing the `touching` block with a `touching color` block (question 5), and then changing the field of the `touching color` block (question 6). Students are then guided to the `when green flag clicked` event of the Toucan Float sprite via another **SE** scaffold and asked to change the fields of a `move` block as well as insert a new `point in direction` block (questions 7 and 8).

In this project's Modify step, students are asked to choose different float costumes, stop the floats at different locations, change the speed of the float movement, and add `say` blocks. The Create step is much more open-ended and has the students create projects based on design prompts. If students have extra time, extension activities are provided including adding other sprites, having sprites perform repeated actions until a condition becomes true, and adding sounds.

4 METHODS

To examine the behaviors of middle-grade students when using the TIPP&SEE learning strategy, we analyzed the completed work of 4th - 7th grade students using the Blinded Curriculum. In the following section, we provide background on the participants within our study, data collection, and data analysis.

4.1 Recruitment and Participants

This study took place in classrooms using the Blinded Curriculum in a large, urban school district in the United States. In total, eight teachers participated in the study. Teachers were selected based on their interest, number of students taught (preference was given to teachers with a larger number of students), and grade levels taught (5th - 7th grades preferred). Altogether, the teachers in this study taught 14 classes and 317 students within those classes consented to being part of the research project. We received data from 307 of the students who consented. All participating teachers and students provided informed consent, parental consent, and/or assent as was appropriate based on age and this research was approved by the lead university's Institutional Review Board and the school district's Research Review Board. Due to the flexibility of the curriculum and school closure due to COVID-19, classes within the study completed

different numbers of modules. The number of students who completed each TIPP&SEE activity ranged from 67 to 147. Additional information about the student and teacher participants can be found in Table 2.

					Yes	**No**
Does your child usually struggle to finish their math homework?					50 (30.67%)	113 (69.33%)
Has your child done any computer programming classes in school?					94 (60.65%)	61 (39.35%)
Has your child participated in computer programming during camps or after-school activities?					35 (21.88%)	125 (78.13%)
Is your child of Hispanic, Latino, or Spanish origin?					69 (38.12%)	112 (61.88%)
Age	9	10	11		12	13
	8 (4.32%)	106 (57.30%)	29 (15.68%)		38 (50.54%)	4 (2.16%)
Gender	Female	Male	Non-Binary/Third Gender			
	98 (53.55%)	85 (46.45%)	0 (0.00%)			
Race	Asian	Native Amr.	Black	Pacific Isl.	White	Other
	15 (7.89%)	4 (2.11%)	68 (35.79%)	1 (0.53%)	68 (35.79%)	4 (17.89%)

Table 1. Student Demographics

4.2 Data Collection

We collected several types of data for this research including a survey about students' backgrounds, completed TIPP&SEE worksheets, and student-created Scratch projects. Our first data source is a background survey. Families were asked to complete the background survey at the beginning of the study and provide information including age, race, gender, experience with mathematics, and previous experience with computing. The survey was completed by the student or caregiver and returned to the research team through the student's teacher. Our second data source is the TIPP&SEE worksheets completed by students. Participating teachers taught the Blinded Curriculum material within their classes and collected all student work. Worksheets for participating students were shared digitally with the research team or provided in hard copy. Finally, we collected and analyzed student Scratch projects. Teachers created separate Scratch studios within their Scratch classrooms for each assignment and students added their projects to these studios. The public studios were shared with researchers for analysis.

4.3 Data Analysis

Our analysis focused on 3 sources of data: TIPP&SEE worksheets, computational artifacts (Modify and Create projects), and student demographic surveys. The analysis primarily focused on the TIPP&SEE worksheets, with data from

Teacher Letter	Grade	Number of Classes	Total Number of Consented Students
B	4	3	51 (16.56%)
C,D,E,H	5	5	118 (38.31%)
G,H	6	2	44 (14.29%)
A,F,G	7	4	95 (30.84%)
Total			317

Table 2. The number of student and classes in the study organized by grade

computational artifacts and demographic surveys being used to situate emerging findings and gain an understanding of additional factors that might be driving behavior found in worksheet data.

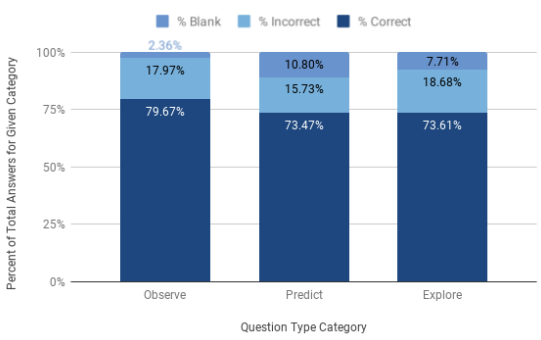
To further contextualize and understand trends observed in the data, all worksheet questions were analyzed in 3 distinct ways. First, each question was categorized based on what students were asked to do and how it aligned to the TIPP&SEE phase, either "Observe", "Predict", or "Explore", described above. Questions were well-defined in one of these categories during the creation of the worksheets and did not require sorting by researchers. Questions that did not fall into any of the categories (such as asking students to report the number of costumes in order to encourage students to discover the costumes tab) were removed from analysis.

The second phase of analysis involved researchers splitting questions into correct and either blank or incorrect answers and extracting trends or patterns.

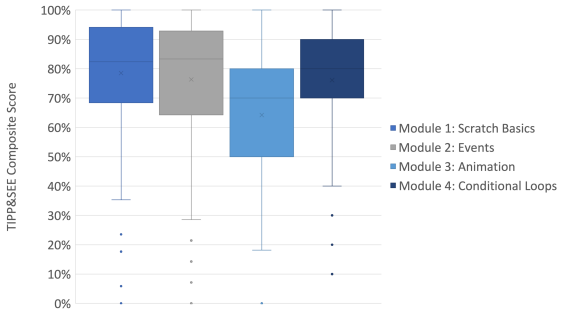
Analysis on where the question was positioned within the worksheet structure was straightforward. This was an objective categorization, with 9 questions per worksheet (3 per section) being labelled as "first", "middle", or "last" within the "Observe", "Predict", and "Explore" sections as described in the previous paragraph.

Completion and accuracy rates of the worksheets were then examined against each of these 3 sets of categorizations to observe whether there was a relationship between student performance on questions and their characteristics. To see if there was any association between student responses and the type of question ("Observe"/"Predict"/"Explore"), and question position, the chi-square test of independence was used. We report χ and p values from this test, with $p < .05$ as our threshold for significance. In addition to characteristics of the questions themselves, we also examined factors related to students and their environment through the demographic survey responses. Specifically, we examined whether students' performance on questions varied depending on whether their parent reported that they had previous programming experience or previously struggled with mathematics homework, both of which were represented as binary "yes" or "no" variables. To compare performance across programming experience, we used the non-parametric Kruskal-Wallis test. That is because some categories of parent-reported programming experience, such as having both school and informal experiences, had very small numbers of students and violated parametric assumptions. We report χ and p values for statistical significance and the η^2 value for effect size. To compare across parent-reported math difficulty, we used the parametric Anova F-test for M1:Basics and M2:Events, which more students completed and whose distribution met the assumptions of the F-test. Its nonparametric analog, the Kruskal-Wallis test, was used for M3:Animation and M4:CondLoops, because of small sample size. We report F, p, and η^2 values from the Anova F-test. We also explored whether a particular teacher or classZroom affected student performance, using the Kruskal-Wallis test because of some small class sizes. Finally, Modify and Create project completion rates were compared to TIPP&SEE worksheet completion rates to assess the degree to which TIPP&SEE encourages creativity and deepens understanding of computational thinking (CT) concepts in middle-grade students.

In order to find out whether worksheet performance correlated with project completion, the programs written in all of the Modify and Create projects were analyzed using static analysis to determine completion rates for a set of predetermined requirements and extension prompts, which were provided on students' project planning worksheets. Examples of these requirements include adding a script for a given sprite, modifying a `say` block for at least 1 sprite, and modifying at least 1 sprite's costume. To extract data from TIPP&SEE worksheets and demographic surveys, which were originally completed by hand, answers were manually transcribed into spreadsheets and checked by multiple researchers. Worksheet answers were then analyzed for completion and accuracy.



(a) Student Performance by Question Type



(b) TIPP&SEE Scores Compared Across Blinded Curriculum Modules

Fig. 4. TIPP&SEE Overall Student Performance Results

5 RESULTS

In this work, we examine the effectiveness of TIPP&SEE with middle-grade students (grades 4-7, ages 9-13). This section begins with an analysis of overall completion and accuracy of the 3 types of questions used in TIPP&SEE. We then investigate potential factors that influence completion and accuracy.

5.1 Overall Performance

We begin by presenting the completion and accuracy broken down into 3 types of questions: "Observe", "Predict", and "Explore".

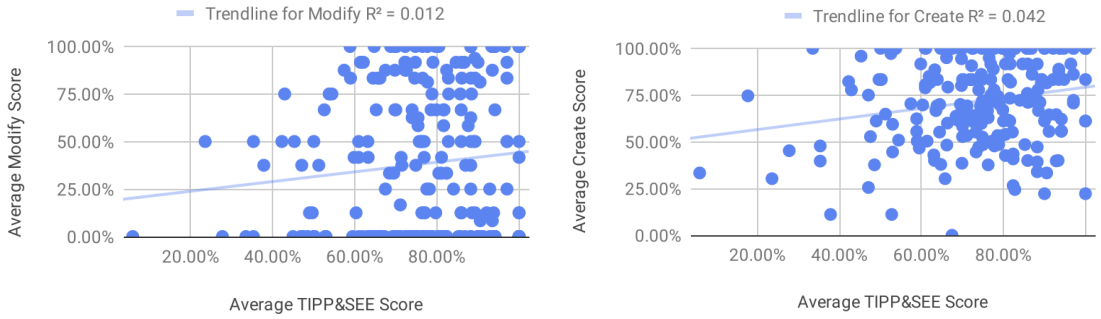
Finding 1: Students have high accuracy, on average, on TIPP&SEE questions. Figure 4a presents the percentages of correct, incorrect, and blank student answers for each worksheet section: "Observe", "Predict", or "Explore". As we can see from the graph, in total, learners answer with 73.61-79.67% accuracy, depending on the question type.

Finding 2: Students attempted the most "Observe" questions, followed by "Explore" questions, and the least "Predict" questions.

There is a statistically-significant association between answer type (if the student answered the question correctly, incorrectly, or not at all) and question type ($\chi = 143.7, p = 2.2 \times 10^{-6}$). Across modules, students attempted questions within each section of the worksheet at different rates. Students attempted a larger percentage of "Observe" questions (which asked students to observe and report what they see happens when they run the code) and "Explore" questions (which asked students to make small modifications to existing code and tell what happens) than when they were asked to hypothesize about the function of a block ("Predict").

Finding 3: Students correctly answered "Observe" questions at a higher rate compared to "Predict" and "Explore" questions.

As shown in Figure 4a, students answered "Predict" and "Explore" questions with comparable correctness (73.47% and 73.61% respectively), while answering "Observe" questions correctly at a higher rate (79.67%). "Explore" questions had the highest rate of incorrect responses (18.68%) compared to 15.73% incorrect for "Predict" questions and 17.67% incorrect for "Observe" questions. The "Observe" questions may have been answered more correctly because students need only view the program to answer them, whereas the "Predict" questions asks students to link behaviors with blocks and the "Explore" questions require students to make changes to the code before answering the questions.



(a) Average Modify scores vs. Average TIPP&SEE scores

(b) Average Create scores vs. Average TIPP&SEE scores

Fig. 5. Average Modify and Create scores vs. Average TIPP&SEE worksheet scores

Finding 4: Performance varied across all modules. Figure 4b shows the performance in each module. While performance was strong in 3 modules, students achieved only 64.20% accuracy, on average, for M3:Animation. Statistical analysis revealed that student worksheet correctness differed across the modules ($F(3, 668) = 12.94, p = 3.18 \times 10^{-8}, \eta^2 = .0549$). Post-hoc analysis indicated that correctness was statistically-significantly different between M3:Animation and M1:Basics ($p = 1 \times 10^{-7}$), M2:Events ($p = 4.7 \times 10^{-6}$), and M4:CondLoops ($p = 6.12 \times 10^{-4}$). This indicates that the technique works overall, but revision may be necessary on that module.

Finding 5: There was little correlation between TIPP&SEE worksheet score and project score.

Figures 5a and 5b display the correlation between the average of students' scores in TIPP&SEE and the average of students' scores in their Modify and Create projects, respectively. The correlation is low for both the Modify ($R^2 = 0.012$) and Create ($R^2 = 0.042$) projects. This result shows there to be little connection between performance on the TIPP&SEE worksheets and performance on the projects. As we will discuss in Section 6.1, this aligns with a different study on 4th-grade students.

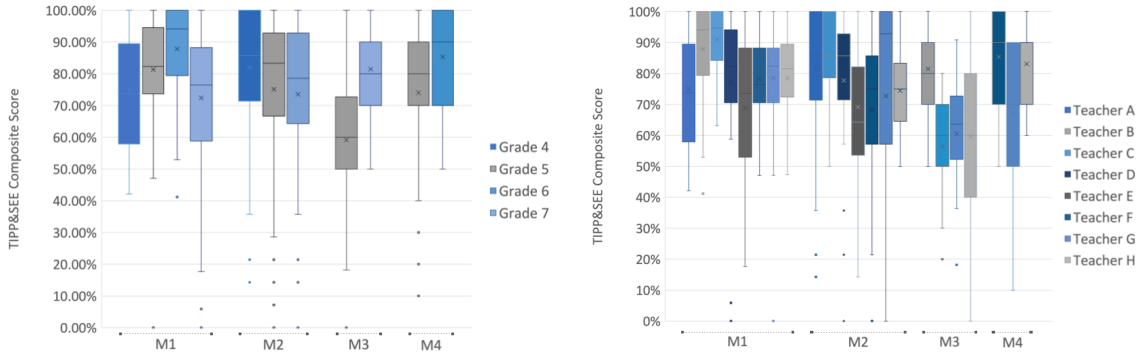
5.2 Potential Performance Influences

While overall performance was good, aggregate performance only tells us how students performed on average and could potentially hide evidence of inequity. That is, an equitable outcome would involve the consistency in performance across students based on a variety of factors. In this section, we explore many different factors that could influence performance.

5.2.1 Classroom-Related Factors. We begin with classroom factors that all students in a classroom share: grade level and teacher.

Finding 6: Grade level was associated with performance for some modules.

Figure 6a illustrates the difference in TIPP&SEE worksheet performance by grade level. For all modules except M2:Events, grade level was linked to worksheet correctness (M1:Basics:$F(3, 255) = 6.67, p < .01, \eta^2 = .0728$; M3:Animation: $\chi = 23.79, p < .01, \eta^2 = .0747$; M4:CondLoops: $\chi = 4.60, p < .05, \eta^2 = .0118$). Post-hoc comparisons for M1:Basics reveal that 6th graders statistically-significantly out-performed both 4th ($p = .0127$) and 7th graders ($p = 9.32 \times 10^{-4}$), and that 5th graders out-performed 7th graders ($p = 8.64 \times 10^{-3}$). It could be that 7th grade students found this Scratch introduction too easy and were not careful in answering the questions. For M3:Animation, the module with the lowest



(a) TIPP&SEE Worksheet Composite Scores of Students in Various Middle-School Grade Levels, by Blinded Curriculum Module

(b) TIPP&SEE Worksheet Composite Scores of Students with Different Teachers, by Blinded Curriculum Module

Fig. 6. TIPP&SEE Scores Compared Across Classroom-Level Factors (Grade and Teacher)

average student performance, 7th graders performed better than 5th graders ($p < .01$). For M4:CondLoops, 6th graders performed better than 5th graders ($p < .05$). This suggests that some modules may be too difficult or too easy for students in specific grade levels and, therefore, more care needs to be taken in tailoring content to different grade levels.

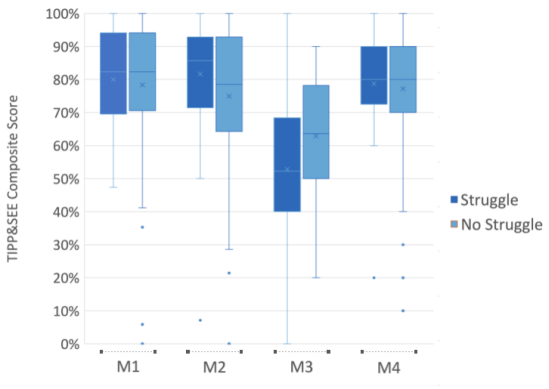
Finding 7: Teacher was associated with performance for all modules. Performance broken down by teacher is depicted in Figure 6b. For all modules, the teacher was significantly linked to worksheet correctness (M1:Basics:$\chi = 38.17, p < .01, \eta^2 = .104$; M2:Events:$\chi = 19.27, p < .01, \eta^2 = .0442$, M3:Animation:$\chi = 29.68, p < .01, \eta^2 = .0881$, M4:CondLoops$\chi = 12.63, p < .01, \eta^2 = .0350$). Teachers are likely to implement the curriculum and learning strategies differently. Additional research is needed to better understand how different teaching techniques influence student performance (on TIPP&SEE worksheets as well as learning).

5.2.2 Student-Related Factors. Next we present results considering relevant academic characteristics of students, namely the effect of their previous math and coding experience on their performance with TIPP&SEE.

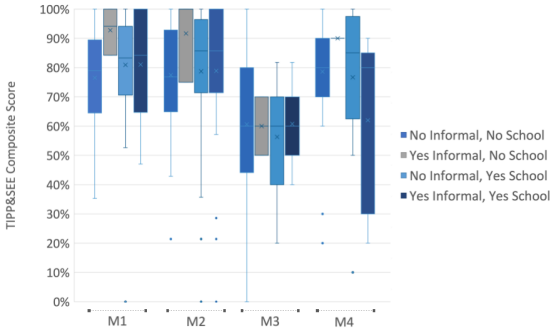
Finding 8: For some modules, students using TIPP&SEE with parent-reported mathematics struggles performed equally well compared to their peers that did not report mathematics struggles and under-performed compared to their peers in others.

Figure 7a presents the average composite TIPP&SEE worksheet scores of students whose parents indicated that the student has struggled to complete their mathematics homework in the past compared to those who did not struggle. The average TIPP&SEE composite score for students with mathematics struggle was 73.50% compared to 73.25% for students with no reported difficulty. Our statistical analysis showed that there was no significant difference in performance on TIPP&SEE worksheets between students with and without parent-reported struggles in math in M2:Events and M4:CondLoops (M2:Events: $F(2, 139) = 1.23, p = 0.295$; M4:CondLoops:$\chi = 0.00367, p = 0.952$). In contrast, students who struggled with mathematics under-performed on M1:Basics and M3:Animation (M1:Basics:$F(2, 147) = 3.17, p < .05, \eta^2 = .0414$; M3:Animation:$\chi = 4.09, p < 0.05, \eta^2 = .0187$). The modules cover different topics and some topics may require a better understanding of math concepts than others.

Finding 9: Students using TIPP&SEE performed equally well on worksheets regardless of their previous experience with programming.



(a) TIPP&SEE Worksheet Composite Scores of Students who do and do not Self-Reportedly Struggle on Mathematics Homework, by Blinded Curriculum Module



(b) Average TIPP&SEE Worksheet Composite Scores of Students with Various Combinations of Levels of Informal and In-School Programming Experience, by Blinded Curriculum Module

Fig. 7. TIPP&SEE Scores Compared Across Levels of Previous STEM Experience

Figure 7b presents the average composite scores of students stratified by their experience with programming in an informal or school setting. While students with only informal experience did slightly better than the others (87.29% vs 76.34%-78.75%), these differences were not statistically-significant (M1:Basics:$\chi = 4.94, p = 0.176$; M2:Events:$\chi = 4.25, p = 0.236$; M3:Animation:$\chi = .820, p = 0.235$; M4:CondLoops:$\chi = 2.19, p = 0.532$). These results are promising and indicate that TIPP&SEE is effective in serving as an equitable learning strategy despite students' previous programming experiences as reported by their parents.

5.2.3 Question-Related Factors. We now delve deeper into individual question differences. The goal was to identify individual questions on which students performed worse than others, identify potential factors that distinguish the poorly-performing from well-performing questions, and perform statistical tests to determine whether that factor correlated with performance. Here, we present the 2 factors we identified that were statistically correlated with differences in performance.

Finding 10: The position of a question within a section was associated with students' attempt and correctness rates.

In Figure 8, we show: correct (percent of attempted questions answered correctly), incorrect (percent of attempted questions answered incorrectly), and blank (percent of questions not attempted) as a function of the position of a question within a section of the worksheet, being the first, middle or last question in a given section. The students tended to attempt fewer questions as they progressed through each section, and the percentage of correct answers decreased through each section ($\chi = 76.06, p < .01$). There was similar percentage of incorrect answers from the first to the last question in the sections.

In the "Observe" section, there were not any noticeable trends in the percentage of correct or incorrect answers from the first question to the last question. However, there was an increase in blank answers from the first to the last question (first: 5.31%, middle: 9.81%, last: 14.50%). Similarly, we did not observe any noticeable trends in the percentage of correct or incorrect answers in the "Predict" sections. However, there was an increase from 5.31% to 14.50% in blank questions for the questions in the "Predict" section.

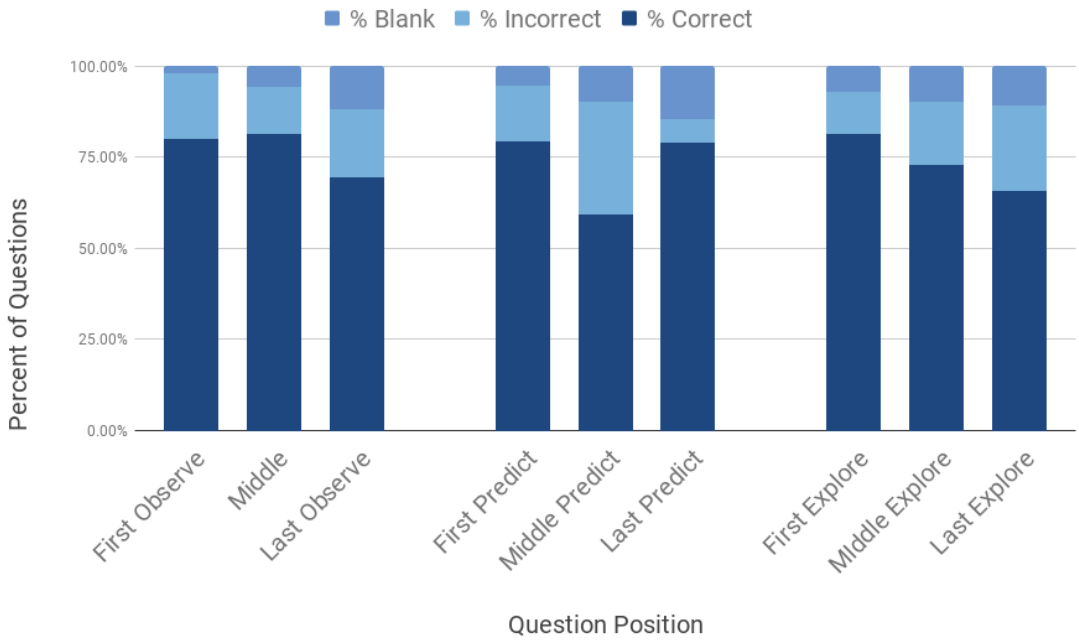


Fig. 8. Question Performance by Question Position in Observe, Predict, and Explore sections on TIPP&SEE Worksheets

The position of a question had the strongest impact on answer correctness in the "Explore" section. For this section, the final section of the worksheets, the percentage of correct answers changed from 81.22% on the first question to 65.83% on the last question. In this section, students incorrectly answered twice as many questions at the end of the section as they did at the start of the section (11.64% to 23.28%). Additionally, there was an increase in blank answers (first: 7.16%, middle: 10.00%, last: 10.94%).

Despite there being no clear trend in incorrectness/correctness with respect to question position in the individual "Observe" and "Predict" sections, the rate of blankness rose steadily from the first to the middle through the last question of all 3 sections. These findings may be due to the students becoming tired of answering questions from the first to last question in the sections.

6 DISCUSSION

In this section, we place our findings in larger context. We first compare to previously-published work - on a different grade level (4th grade only) and a different curriculum - in order to understand what may be a trend related to TIPP&SEE itself vs. what may be correlated with a specific grade level or implementation. Second, we look at findings that were unique to this study to relate them to potential reasons beyond this study.

6.1 Relating to Prior Work

Here, we compare the results in our study with the results of one previously-published study on student behavior using TIPP&SEE [14]. There are 2 major differences between their study and our study. First, their study consisted of only

Comparison	Finding
Similar	Students completed and correctly answered "Observe" questions the most
Dissimilar	Students leave the most blank answers in "Predict" questions in our study and students leave the most blank answers in "Explore" questions in the prior study
Similar	Students answered "Predict" questions incorrectly the least often
Dissimilar	Our study's students had higher *accuracy* on "Explore" questions than prior study
Similar	Completion and accuracy rates differ by teacher
Similar	TIPP&SEE completion and accuracy are not correlated with project completion

Table 3. Findings Comparison

4th-grade classrooms, many of which were bilingual. Second, they used a different curriculum. The curriculum had several differences, including different projects, different formatting of their TIPP&SEE worksheets, different individual questions on the TIPP&SEE worksheets, and slightly different content (no coverage of conditional loops). Our goal is to gain insight into what findings based on the use of TIPP&SEE hold across curricula and age groups.

To perform this comparison, we focus on the findings in both papers and compare which ones are both present and whether they align or are contradictory. The comparison of similar and dissimilar findings are also shown in Table 3

Observe, Predict, and Explore. In both studies, students answered "Observe" questions most often and most accurately. In addition, "Predict" questions were answered incorrectly the least often but were left blank the most. That these trends hold across both studies suggests that these trends are likely inherent in these types of questions.

Thinking about the 3 types of questions, this is perhaps not surprising, because "Observe" questions ask about what they see in a running program, which is analogous to merely using a program. "Predict" and "Explore" questions are inherently more difficult but for different reasons. "Predict" questions are more cognitively difficult, asking about which block performed which action. "Explore" questions, on the other hand, require students to make specific changes and compare the output before the change to after the change. This requires that the student actually perform the change properly and remember the prior output accurately. Students may just attempt to guess what will happen if they do not want to go through the work of changing it.

It is also interesting that the "Explore" questions were attempted more often in this work than the "Predict" questions, which may provide insight into the student learning process. There is perhaps less of a cognitive barrier to changing the code and inspecting the behavior ("Explore") than predicting the outcome without the opportunity to explore the effects of the code change ("Predict"). This may also explain why students answered more "Explore" questions incorrectly than the other questions. We believe that our current results differ from previous results [14] because our dataset includes students from 4th through 7th grade. The older students may become tired less easily and also complete the work more quickly. The younger students in the previous study may have also focused on the coding part of the "Explore" section and forgotten to complete the worksheet part of the section. These findings suggest a need for more guidance and support for the "Predict" and "Explore" questions. Specifically, we need to encourage students to attempt "Predict" questions more. For "Explore" questions, further examination is needed to see why their answers are incorrect more often.

TIPP&SEE and Projects. Similar to findings from previous work [14], our analysis revealed the correlation between TIPP&SEE worksheet score and project score was weak.

When we take the finding that teacher influences worksheet completion and accuracy and the finding that worksheets do not correlate with project score together, and triangulate that with our observation field notes, a possible theory

emerges. Different teachers place different emphasis on accurate completion of the worksheet. One teacher, for example, had students check off their completed worksheet with the teacher. If it was not correct or complete, the teacher required that they go back and try again until it was correct. That teacher did not allow students to continue to modify without accurately completing the TIPP&SEE. Another teacher, on the other hand, went over the worksheet with the entire class before moving on. Both techniques are based on sound pedagogical principles - they wanted to make sure students had some basic understanding of the TIPP&SEE exercise prior to moving on. In both classrooms, we would expect to see higher completion and accuracy than other classrooms. In addition, the first teacher's technique may lead to better learning since students needed to figure out the answers themselves rather than being told by the teacher later. Because teachers varied in how they treated the worksheet, it is not unexpected that the completion and accuracy differed. Furthermore, these differences in teachers' strategies could also affect whether or not there is a correlation with student understanding and/or project performance.

6.2 Unique Findings

We now discuss findings that were not in prior work in order to explore possible explanations and implications.

Question Location. Students tended to attempt fewer questions, and answer them with decreasing accuracy, as they progressed through the worksheet. This was most dramatic in the final section of the worksheet. These findings could be due to cognitive fatigue [40]. That is, students may become fatigued from answering questions from the beginning of a section to the end of a section, and from the beginning of a worksheet to the end of a worksheet. In order for students to have the desired learning outcomes, we must further understand the sources of worksheet fatigue and work to mitigate these effects where possible. This might entail worksheets of varying length and providing an option to pause and restart a given worksheet at a later time, along with a study to see if such methods improve learning fatigue.

Math and Programming. There were not any differences in student outcomes based on previous experience with programming, indicating TIPP&SEE may have served effectively as an equitable factor across differences in parent-reported previous programming experience. This further bolsters the argument to use TIPP&SEE in classroom environments where students have varying levels of programming experience and are not able to receive highly personalized attention from educators.

Results were more mixed with respect to parent-reported mathematics struggles, where students with math struggles performing similarly to their peers without math struggles in some modules and under-performing in other modules. This echoes prior work that has found differences in CS assessment performance middle-school students [19, 35] of different math levels, with differences becoming more prevalent in more advanced CT concepts. However, we do have to keep in mind that our data was collected from parental reports instead of more objective assessments. Due to TIPP&SEE's abilities to support students regardless of prior programming experience and support students who struggle with mathematics in some modules, TIPP&SEE should be used with middle-grade students. However, we should increase the support in modules where TIPP&SEE was unable to equalize students who struggle with math with students who do not.

Module and Teacher. We found that across 2 modules, student performance with TIPP&SEE did not significantly vary based on grade level. However, their performance did vary between modules and between teachers. Because of TIPP&SEE's similar effectiveness across grade levels in middle-grade students, it may be particularly adept for environments where only one teacher is responsible for teaching computing concepts to students from several different grades.

We also analyzed the students' performance at various grade levels. First, we found that 6th graders differed from 4th and 7th in M1:Basics We believe that this may be the case because the 6th grade teachers may have been more experienced. Additionally, 5th graders differed from 7th graders in M1:Basics. We also found that 7th graders differed from 5th graders in M3:Animation and 6th graders differed from 5th graders in M4:CondLoops. This suggests that 5th graders may need more support in some modules and that even though students in all grade levels may be able to complete TIPP&SEE, they may still struggle with answering questions correctly. Tied to that finding is our finding that performance differed between modules. M3:Animation had the lowest average TIPP&SEE scores. This further supports the idea that modules vary in difficulty and students also need varying support.

7 LIMITATIONS

The data collection was interrupted by the COVID-19 pandemic and the subsequent switch to online learning. Classroom Blinded Curriculum instruction ended in many cases when classes unexpectedly switched to an online format. The number of questions presented to students and their answers overall is large, but each class size and the number of students who were given each specific worksheet varied. This is due to that different classes completed varying numbers of modules and fewer classes completed Modules 3 and 4. Finally, there are limitations involved in the demographic survey in which parents reported if their student struggled to complete their mathematics homework or if they have experience in programming. Parent answers may be exaggerated, the parents may feel embarrassed to reveal if their student struggles in school, and students may feel embarrassed to reveal to their parents if they struggle in school.

7.1 Future Work

During our analysis, we identified particular questions which were frequently answered incorrectly or not answered at all. While we believe some of this could be explained by our finding that the position of a question is associated with students' performance on it, we also believe more work can be done to investigate additional facets driving students' behavior with respect to these questions.

We speculate there were two common themes among such more difficult questions: 1) their answers choices could potentially be misinterpreted by students attempting to apply common-sense reasoning from their day-to-day lives to answer the questions without completing all required steps, such as guessing what would happen in an "Explore" question without actually making the changes and 2) student may struggle with interpreting questions requiring advanced mathematical concepts such as the Cartesian coordinate system or degree angle measurements. As such, future analysis could be conducted whereby "Predict" and "Explore" questions (that is, those requiring students to deal directly with the code) could be stratified into categories based on "everyday" or "mathematic" knowledge. Completion and accuracy within each of these segments could then be analyzed to determine whether students performed statistically significantly better or worse on questions fitting these criteria. Concrete findings emerging from such work in the future would enable us to design curricula which provide additional scaffolding for potentially out-of-scope mathematical questions.

8 CONCLUSIONS

As CS education becomes accessible to more middle-school students, it is important that we closely examine instructional methods for their use across several grades. In this paper, we explore the TIPP&SEE strategy and its enactment in Blinded Curriculum. We find that, for the most part, the trends observed at the 4th-grade level in a different curriculum held with this middle-grade implementation: students complete "Observe" questions most often and accurately, teachers

are correlated with worksheet performance, and worksheet performance is not correlated with project performance. This study also had additional findings, such as that students' prior experience with programming had no effect on worksheet performance, and math struggles only affected performance for some modules. We also discovered that students lose engagement and accuracy as they progress toward the end of a TIPP&SEE worksheet.

9 ACKNOWLEDGEMENTS

We would like to thank the teachers and students who piloted Blinded Curriculum. This material is based upon work supported by [Blinded for review]

REFERENCES

[1] authors. [n. d.].

[2] Karen Brennan and Mitchel Resnick. 2012. 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. 25.

[3] Ann L Brown, Annemarie S Palincsar, and Bonnie B Armbruster. 1984. Instructing comprehension-fostering activities in interactive learning situations. *Learning and comprehension of text* (1984), 255–286.

[4] Feryal Cubukcu. 2008. Enhancing vocabulary development and reading comprehension through metacognitive strategies. *Issues in Educational Research* 18, 1 (2008), 1–11.

[5] Susan De La Paz. 2005. Effects of historical reasoning instruction and writing strategy mastery in culturally and academically diverse middle school classrooms. *Journal of Educational Psychology* 97, 2 (2005), 139.

[6] Donald D Deshler and Jean B Schumaker. 1986. Learning strategies: An instructional alternative for low-achieving adolescents. *Exceptional Children* 52, 6 (1986), 583–590.

[7] Anouk S Donker, Hester De Boer, Danny Kostons, CC Dignath Van Ewijk, and Margaretha PC van der Werf. 2014. Effectiveness of learning strategy instruction on academic performance: A meta-analysis. *Educational Research Review* 11 (2014), 1–26.

[8] S Dymock. 2005. Teaching Expository Text Structure Awareness. *The Reading Teacher* 59, 2 (2005), 117–181.

[9] I. Lee et al. 2011. Computational thinking for youth in practice. *ACM Inroads* 2, 1 (2011), 32–37.

[10] Sharon Vaughn et al. 2011. Efficacy of Collaborative Strategic Reading With Middle School Students. *American Education Research Journal* 48, 4 (2011), 938–964.

[11] Katrina Falkner, Sue Sentance, Rebecca Vivian, Sarah Barksdale, Leonard Busuttil, Elizabeth Cole, Christine Liebe, Francesco Maiorana, Monica M. McGill, and Keith Quille. 2019. An International Study Piloting the MEasuring TeacheR Enacted Computing Curriculum (METRECC) Instrument. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education (ITiCSE-WGR '19)*. Association for Computing Machinery, New York, NY, USA, 111–142. https://doi.org/10.1145/3344429.3372505

[12] Diana Franklin, Merijke Coenraad, Jennifer Palmer, Donna Eatinger, Anna Zipp, Marco Anaya, Max White, Hoang Pham, Ozan Gökdemir, and David Weintrop. 2020. An Analysis of Use-Modify-Create Pedagogical Approach's Success in Balancing Structure and Student Agency. In *Proceedings of the 2020 ACM Conference on International Computing Education Research*. 14–24.

[13] Diana Franklin, Jean Salac, Zachary Crenshaw, Saranya Turimella, Zipporah Klain, Marco Anaya, and Cathy Thomas. 2020. Exploring Student Behavior Using the TIPPamp;SEE Learning Strategy. In *Proceedings of the 2020 ACM Conference on International Computing Education Research (ICER '20)*. Association for Computing Machinery, New York, NY, USA, 91–101. https://doi.org/10.1145/3372782.3406257

[14] Diana Franklin, Jean Salac, Zachary Crenshaw, Saranya Turimella, Zipporah Klain, Marco Anaya, and Cathy Thomas. 2020. Exploring Student Behavior Using the TIPP&SEE Learning Strategy. In *Proceedings of the 2020 ACM Conference on International Computing Education Research*. 91–101.

[15] Diana Franklin, David Weintrop, Jennifer Palmer, Merijke Coenraad, Melissa Cobian, Kristan Beck, Andrew Rasmussen, Sue Krause, Max White, Marco Anaya, and Zachary Crenshaw. 2020. Scratch Encore: The Design and Pilot of a Culturally-Relevant Intermediate Scratch Curriculum. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 794–800.

[16] Ursula Fuller, Colin G. Johnson, Tuukka Ahoniemi, Diana Cukierman, Isidoro Hernán-Losada, Jana Jackova, Essi Lahtinen, Tracy L. Lewis, Donna McGee Thompson, Charles Riedesel, and Errol Thompson. 2007. Developing a Computer Science-Specific Learning Taxonomy. *SIGCSE Bull.* 39, 4 (Dec. 2007), 152–170. https://doi.org/10.1145/1345375.1345438

[17] Russell Gersten, Lynn S Fuchs, Joanna P Williams, and Scott Baker. 2001. Teaching reading comprehension strategies to students with learning disabilities: A review of research. *Review of educational research* 71, 2 (2001), 279–320.

[18] Steve Graham, Debra McKeown, Sharlene Kiuhara, and Karen R Harris. 2012. A meta-analysis of writing instruction for students in the elementary grades. *Journal of educational psychology* 104, 4 (2012), 879.

[19] Shuchi Grover, Roy Pea, and Stephen Cooper. 2016. Factors influencing computer science learning in middle school. In *Proceedings of the 47th ACM technical symposium on computing science education*. 552–557.

[20] I. Harel and S. Papert. 1990. Software design as a learning environment. *Interactive Learning Environments* 1, 1 (1990), 1–32.

[21] Alison King. 1991. Improving lecture comprehension: Effects of a metacognitive strategy. *Applied Cognitive Psychology* 5, 4 (1991), 331–346.

[22] Bracha Kramarski, Zemira R Mevarech, and Marsel Arami. 2002. The effects of metacognitive instruction on solving mathematical authentic tasks. *Educational studies in mathematics* 49, 2 (2002), 225–250.

[23] Irene Lee, Fred Martin, Jill Denner, Bob Coulter, Walter Allan, Jeri Erickson, Joyce Malyn-Smith, and Linda Werner. 2011. Computational thinking for youth in practice. *Acm Inroads* 2, 1 (2011), 32–37.

[24] Nicholas Lytle, Veronica Cateté, Danielle Boulden, Yihuan Dong, Jennifer Houchins, Alexandra Milliken, Amy Isvik, Dolly Bounajim, Eric Wiebe, and Tiffany Barnes. 2019. Use, Modify, Create: Comparing Computational Thinking Lesson Progressions for STEM Classes. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*. 395–401.

[25] John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. 2010. The Scratch Programming Language and Environment. *Trans. Comput. Educ.* 10, 4, Article 16 (Nov. 2010), 15 pages.

[26] Siti Nurulain Mohd Rum and Maslina Zolkepli. 2018. Metacognitive Strategies in Teaching and Learning Computer Programming. *International Journal of Engineering Technology* 7 (12 2018), 788. https://doi.org/10.14419/ijet.v7i4.38.27546

[27] Marjorie Montague. 1992. The effects of cognitive and metacognitive strategy instruction on the mathematical problem solving of middle school students with learning disabilities. *Journal of learning disabilities* 25, 4 (1992), 230–248.

[28] N. S. Nasir, A. S. Rosebery, B. Warren, and C. D. Lee. 2006. *Learning as a cultural process: Achieving equity through diversity.* Cambridge Univ Pr.

[29] Richard Noss and Celia Hoyles. 1996. *Windows on mathematical meanings: Learning cultures and computers*. Vol. 17. Springer Science & Business Media.

[30] S. Papert. 1980. *Mindstorms: Children, Computers, and Powerful Ideas.* Basic Books, Inc.

[31] Seymour Papert. 1993. *The children's machine: Rethinking school in the age of the computer.* ERIC.

[32] Seymour Papert and Idit Harel. 1991. Situating constructionism. *Constructionism* 36, 2 (1991), 1–11.

[33] S. Papert, D. watt, A. diSessa, and S. Weir. 1979. *Final report of the Brookline Logo Project: Project summary and data analysis (Logo Memo 53).* Technical Report. MIT Logo Group, Cambridge, MA.

[34] Scott G Paris, Peter Winograd, et al. 1990. How metacognition can promote academic learning and instruction. *Dimensions of thinking and cognitive instruction* 1 (1990), 15–51.

[35] Yizhou Qian and James D Lehman. 2016. Correlates of Success in Introductory Programming: A Study with Middle School Students. *Journal of Education and Learning* 5, 2 (2016), 73–83.

[36] Jean Salac, Cathy Thomas, Chloe Butler, and Diana Franklin. 2021. Supporting Diverse Learners in K-8 Computational Thinking with TIPPSEE *(SIGCSE '21)*. Association for Computing Machinery, New York, NY, USA, 246–252. https://doi.org/10.1145/3408877.3432366

[37] Jean Salac, Cathy Thomas, Chloe Butler, Ashley Sanchez, and Diana Franklin. 2020. TIPP&SEE: A Learning Strategy to Guide Students through Use–>Modify Scratch Activities. In *Proceedings of the 2019 ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE '20)*. ACM.

[38] Jean Salac, Cathy Thomas, Chloe Butler, Ashley Sanchez, and Diana Franklin. 2020. TIPPSEE: A Learning Strategy to Guide Students through Use->Modify Scratch Activities. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA.

[39] Thomas E Scruggs, Margo A Mastropieri, Sheri L Berkeley, and Lisa Marshak. 2010. Mnemonic strategies: Evidence-based practice and practice-based evidence. *Intervention in School and Clinic* 46, 2 (2010), 79–86.

[40] Hans Henrik Sievertsen, Francesca Gino, and Marco Piovesan. 2016. Cognitive fatigue influences students' performance on standardized tests. *Proceedings of the National Academy of Sciences* 113, 10 (2016), 2621–2624.

[41] Lev Semenovich Vygotsky. 1980. *Mind in society: The development of higher psychological processes.* Harvard university press.

[42] David Weintrop. 2019. Block-based Programming in Computer Science Education. *Commun. ACM* 62, 8 (July 2019), 22–25. https://doi.org/10.1145/3341221

[43] David Weintrop, David C Shepherd, Patrick Francis, and Diana Franklin. 2017. Blockly goes to work: Block-based programming for industrial robots. In *2017 IEEE Blocks and Beyond Workshop (B&B)*. IEEE, 29–36.

[44] David Weintrop and Uri Wilensky. 2015. To Block or Not to Block, That is the Question: Students' Perceptions of Blocks-based Programming. In *Proceedings of the 14th International Conference on Interaction Design and Children (IDC '15)*. ACM, New York, NY, USA, 199–208.

[45] David Wood, Jerome S Bruner, and Gail Ross. 1976. The role of tutoring in problem solving. *Journal of child psychology and psychiatry* 17, 2 (1976), 89–100.