Building Upon and Enriching Grade Four Mathematics Standards with Programming Curriculum

Colleen M. Lewis
Graduate School of Education
University of California, Berkeley
Berkeley, CA 94720
colleenl@eecs.berkeley.edu

Niral Shah
Graduate School of Education
University of California, Berkeley
Berkeley, CA 94720
niralshah@gmail.com

ABSTRACT
We found that fifth grade students' scores on Scratch programming quizzes in a summer enrichment course were highly correlated with their scores on a standardized test for mathematics. We identify ways in which the programming curriculum builds upon target skills from the *Mathematics Content Standards for California Public Schools*. We hypothesize that the programming curriculum leveraged and enriched students' mathematics content knowledge.

Categories and Subject Descriptors
K.3.2 **[Computers and Education]:** Computer and Information Science Education—*computer science education*

General Terms
Human Factors

Keywords
Scratch, middle-school, mathematics, computational thinking

1. INTRODUCTION
Methods for predicting college students' success in a programming course have been hotly debated for decades (see [15] for a review). The current study investigated correlations between young students' standardized test scores and their performance in a programming course. However, rather than aiming to predict success, the intention of the present study is to uncover dependencies between a middle school programming curriculum in a language called Scratch and grade-specific mathematical content.

We found students' scores on the Grade 4 California Mathematics Standards Test predicted performance on programming quizzes in a summer enrichment program, which was statistically significant at the 1% level. Our hypothesis that students' scores on the Grade 4 California English-Language Arts Standards Test would be correlated with performance on programming quizzes was not confirmed.

2. PREVIOUS RESEARCH
For decades, research has attempted to correlate students' non-programming and programming performance. Simpson [16] in 1973 published a bibliography of more than 150 papers, which attempted to identify methods for selecting computing staff. In summarizing the set of papers, Simpson noted that: "They mainly report on correlations of test scores with either course grades or supervisor ratings and almost all conclude that further research is required." (p. 2) [16]. For example, in 1985 Butcher and Muth [2] developed a model to attempt to predict performance in a computing course based upon ACT test scores. They reported a high level of false negatives; approximately one-third of the students that they predicted would not be successful in the undergraduate program were in fact successful.

In 2006 Simon *et al.* [15] reported that "the results achieved by students in programming courses do not correlate well with their other academic results." (p. 190) [15]. Further, Rountree *et al.* argued that "despite the attention given to this topic, a reliable means of predicting the success of students who enter an introductory programming course remains elusive." (p. 101) [13].

This long line of research opens up questions regarding the complexity of transferring knowledge between non-programming and programming domains, which is part of our ongoing research program [21]. However, the current paper does not seek to predict performance in a computing course. We hypothesize that mathematical content standards in the early grades are more aligned with the mathematical content embedded within our programming curriculum and seek to investigate this hypothesis. Based upon this hypothesis we predict that students' performance on mathematics standardized tests in elementary school would be correlated with performance on programming quizzes.

A second feature that separates the current work from related correlational research is that, as others note [12], [21], much of previous work has looked for predictors of programming aptitude, assuming the existence of an innate ability for computer science. But we reject this assumption regarding innate ability as unsubstantiated and reiterate its potentially negative effects on student learning. Of greater relevance is previous research using programming to support content learning in another domain.

The most recent and visible example of using programming as a tool for mathematics learning is the Bootstrap project [14] [19]. This project uses a functional programming environment to teach game development and programming to students as young as 11 [14]. The curriculum is designed to introduce and support the development of students' competencies in "algebra, coordinate geometry, and simple modeling" [19]. Unfortunately, there appear to be no studies to date assessing students' transfer of this mathematical content knowledge between the programming and non-programming domains.

The Bootstrap project builds upon earlier examples where programming was used as a tool for mathematics learning. In the 1980's Sutherland [17] and Noss [10] showed students could develop conceptual understanding of variables through

SIGCSE '12, February 29 –March 3, 2012, Raleigh, North Carolina, USA.

programming in Logo. In another example from 1992, Harel & Papert [8] were successful in supporting the learning of fractions in a Logo programming environment with fourth grade students. Students in the treatment group learned Logo with the goal of developing software to teach younger children about fractions. Students in the control group spent the same amount of time learning Logo and completed the same two-month mathematics unit on fractions. However, students in the treatment group demonstrated greater mastery of both the Logo and fractions content.

These previous research projects are examples of a broader trend of using programming "microworlds" [11] to explore non-programming content. For example, Abelson and diSessa [1] authored a textbook using Logo programming as a tool for teaching introductory and college level geometry. diSessa [7] mapped the ways in which programming microworlds could empower new ways of learning and knowing physics. diSessa [7] describes the general idea of "computational literacy", a precursor to "computational thinking" [18], in which programming can function as a new literacy (like algebra), reshaping scientific communication. As diSessa claims: "Not only can new inscription systems and literacies ease learning, as algebra simplified the proofs of Galileo's theorems, but they may also rearrange the entire intellectual terrain." (p. 19) [7]. We build upon this existing body of work to support non-programming content learning using programming as a medium.

3. METHODS

3.1 Classroom Context

Forty-seven students in the summer of 2011 were enrolled in a summer enrichment program for students entering the sixth grade and agreed to participate in a research study. Of these students, approximately 30% were female. These students were divided between two classes that were co-taught by the authors, each of whom have a degree in computer science. One of the instructors had taught the course twice in the same enrichment program [20] [22]. The other instructor had five years of experience teaching high-school math, but had never taught programming.

The course introduced programming in Scratch [20], Snap [9], and Logo [8], with the majority of 36 hours of class spent using Scratch. Although students frequently drew on their mathematical knowledge, the course was designed to teach students programming constructs from these languages, and was not specifically designed around mathematics content. Students programmed on their own computers. Each day, students were assigned a partner and were expected to discuss any questions with their partner before asking an instructor [22]. Details regarding the curriculum (available at www.colleenmlewis.com/scratch/) and pedagogical techniques developed for teaching young students programming can be found in a previous publication regarding the course [22].

3.2 Sources of Data

3.2.1 Standardized Test Scores

Participants in the research were given the option to submit the numerical scores for a recent standardized test given in the state of California (CA). Students' most recent CA state test was taken during their fourth grade year, in April of 2010. The tests are designed to assess students' performance relative to the state's content standards for the particular grade. Students receive a score between 150 (low) to 600 (high) for the Mathematics and English-Language Arts component separately.

A score below 300 is described as "Below Basic" and demonstrates "little or flawed understanding" for the content area assessed. A score between 300 and 349 is described as "Basic" and demonstrates "partial and rudimentary understanding". A score between 350 and 400 (or 392 for English-Language Arts) is described as "Proficient" and demonstrates "competent and adequate understanding". Above this score is described as "Advanced" and demonstrates "comprehensive and complex understanding."

Twenty-two of the forty-seven students who agreed to participate in the study submitted test scores. The summer enrichment program was designed for academically advanced students and the submitted test scores show their performance was well above the state average. On the 2010 administration, only 42% and 36% of fourth graders in California received a score of "Advanced" in Mathematics and English-Language Arts respectively [4]. In contrast, in the study population, 100% and 95% of students received a score of "Advanced" for Mathematics and English-Language Arts respectively. Figure 1 shows a scatter plot for the scores submitted by study participants. In our sample, 95% of students had an equal or higher score for Mathematics than English-Language Arts. Figure 2 shows box plots of the two sets of scores.

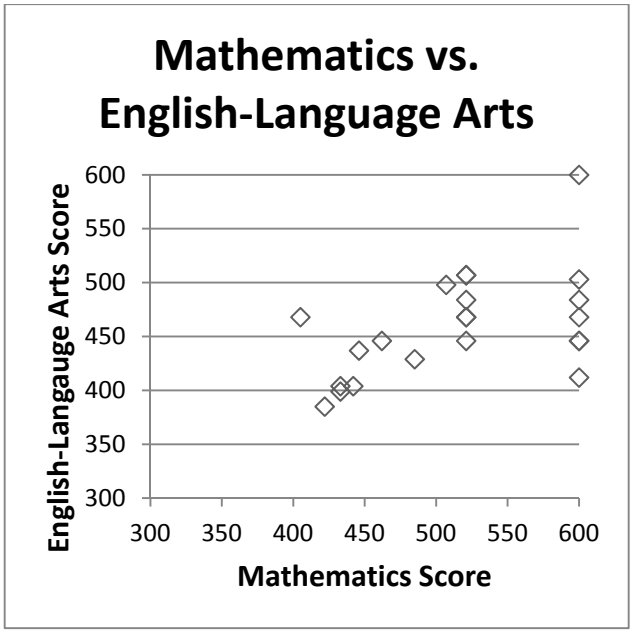

Figure 1. Scatter plot of Mathematics and English-Language Arts scores from participants' Grade 4 CA Standards Tests.

3.2.2 Programming Quizzes

The 36 hour programming course occurred on 12 days during the summer of 2011. On the 2nd through 10th days of instruction, students took paper-based quizzes assessing their knowledge of the programming content from the previous day. With the exception of one quiz, the questions were consistent between this study and a previous study [22], which includes a representative question from each quiz. Quiz questions were graded with one point for most questions; no partial credit was given. Each quiz had between 11 and 19 points, and each quiz point was weighted equally when calculating students' average quiz performance. Quizzes lasted on average 15 minutes and there were a total of 148 points possible across all quizzes.

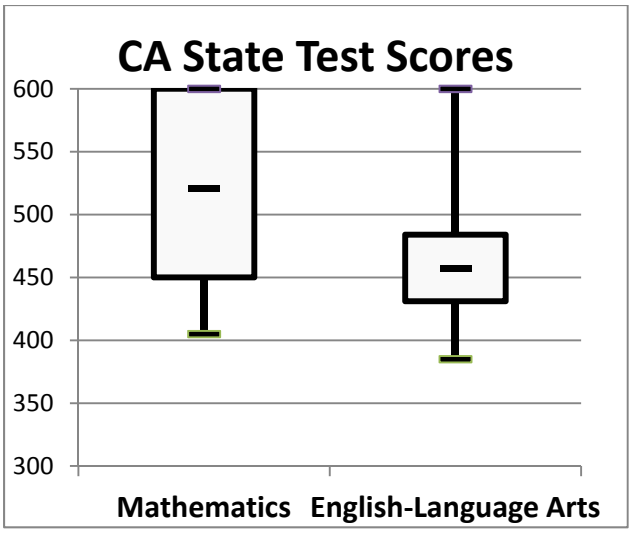

Figure 2. Box-plot of Mathematics and English-Language Arts scores from participants' Grade 4 CA Standards Tests.

The majority of the questions appearing on the programming quizzes had mathematical content, identified as the use of variables, mathematical computation, or geometric relationships. We hypothesized that questions without such mathematical content may have a different pattern of correlation with students' mathematics scores. A portion of one such non-mathematical question is shown in Figure 3, which asks students to identify scripts that "work like this one". The question also included six additional scripts, two of which are equivalent to the first script shown in Figure 3. For reference, two incorrect answers are shown below in Figure 3.

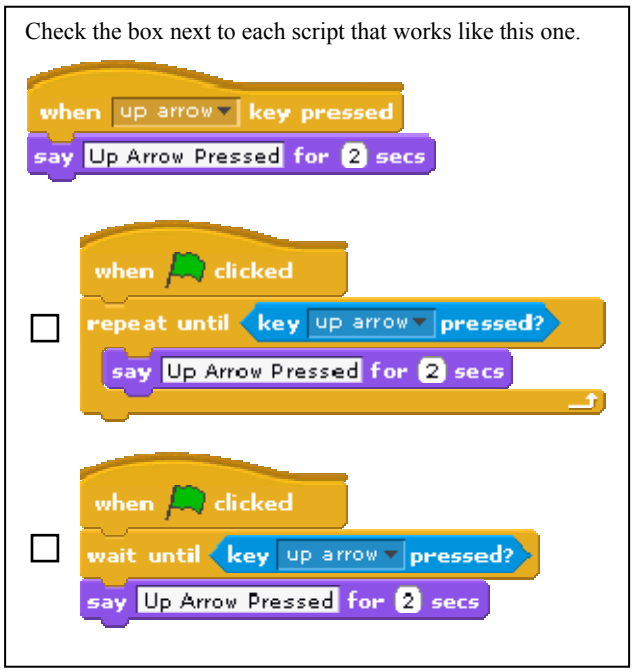

Figure 3. Question taken from the 9th programming quiz with two incorrect answers.

3.3 Analytical Methods

This study employed both quantitative and qualitative methods for interpreting patterns in students' CA state test scores and programming quiz performance. We investigated three hypotheses using t-tests and a standard statistical significance level of 0.05.

- Are students' scores on programming quizzes correlated with their reported scores on the CA Grade 4 Mathematics Test?
- Are students' scores on programming quizzes correlated with their reported scores on the CA Grade 4 English-Language Arts Test?
- When controlling for students' English-Language Arts scores, are students' scores on non-mathematical components of the programming quizzes correlated with their reported scores on the CA Grade 4 Mathematics Test?

To better understand the data we identified overlap between content covered on the Grade 4 California Mathematics Standards and the programming content of daily quizzes. This additional qualitative analysis of the mathematics standards [5] and example questions [3] may provide insight regarding the results of the statistical tests.

4. QUANTITATIVE RESULTS

We found that students' test scores in mathematics were highly correlated with performance on the programming quizzes in the summer enrichment course (t=3.461, p=0.0025), significant at the 1% level. The line of best fit shows that for a 50 point increase in mathematics score (ranging from 150 to 600) the student will answer correctly an additional 20 points out of the 148 total quiz points. Figure 4 shows a scatter plot and line of best fit comparing the 22 students' scores in mathematics and their average quiz performance in the summer enrichment program after completing the fifth grade.

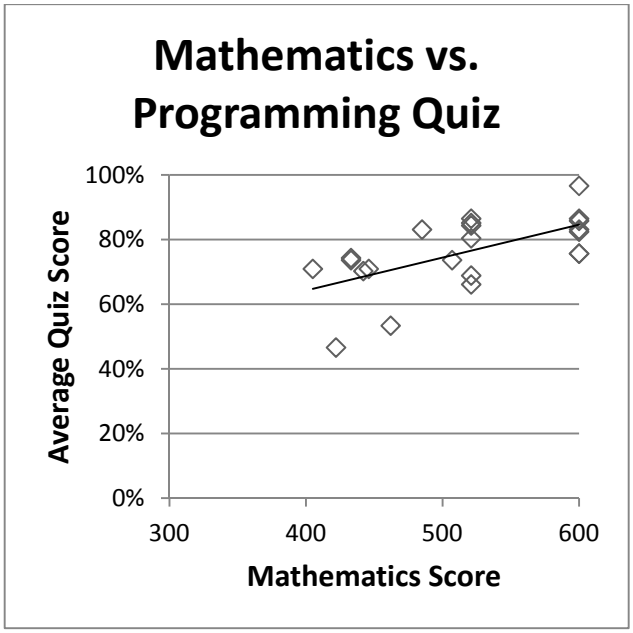

Figure 4. Scatter plot of students' Mathematics scores versus average quiz performance.

Testing the second hypothesis, we could not reject the null hypothesis that students' scores on the English-Language Arts component of their grade four test were not correlated with their performance on quizzes within the enrichment program (t = 1.288, p=0.213). Figure 5 shows a trend for higher English-Language Arts scores to be associated with higher average quiz performance; however this trend is not statistically significant at

the 5% level. When both the Mathematics and English-Language Arts components are included in the linear model, the statistical significance of the English-Language Arts component drops (t = -0.217, p = 0.830), while the Mathematics component remains significant at the 1% level (t = 3.020 , p = 0.007).

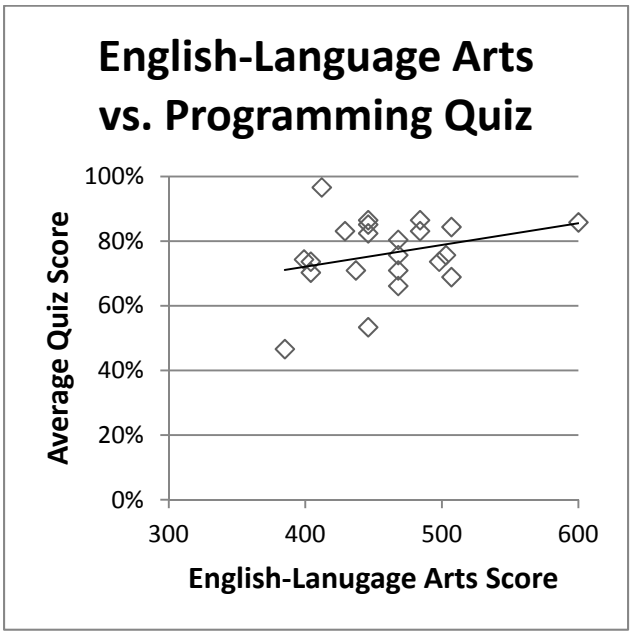

Figure 5. Scatter plot of students' English-Language Arts scores versus average quiz performance.

Investigating the third hypothesis, we found that when controlling for students' scores on the English-Language Arts component, the Mathematics scores were not a statistically significant predictor of performance on non-mathematical quiz questions (t = 1.357, p = 0.191), such as the question shown in Figure 3.

5. QUALITATIVE ANALYSIS: MATHEMATICS CONTENT STANDARDS

The Grade 4 California Mathematics Standards are summarized as: "By the end of grade four, students understand large numbers and addition, subtraction, multiplication, and division of whole numbers. They describe and compare simple fractions and decimals. They understand the properties of, and the relationships between, plane geometric figures. They collect, represent, and analyze data to answer questions." [5].

In grade four students are assessed within five content areas, "Number Sense," "Algebra and Functions," "Measurement and Geometry," "Statistics, Data Analysis, and Probability," and "Mathematical Reasoning." These content areas are subdivided into a total of 14 sub-areas, which are further subdivided into 54 standards. The authors estimate that more than 35% of the CA state standards for grade four apply to the curriculum used in our 36-hour summer enrichment program. We expect that in these cases the curriculum would both build upon the students' mathematical content knowledge and reinforce or enrich that knowledge.

As it is not possible in this venue to identify every instance of overlap within the 54 content standards for grade four, we provide three representative examples comparing questions taken from a set of released test questions for Grade Four Mathematics [3] and questions from the programming quizzes.

5.1 Example 1: Algebra and Functions

The question shown in Figure 6 was taken from a previous Grade 4 CA Mathematics Standards Test. It exemplifies the following content standard within the "Algebra and Functions" content area (Standard #1.5): "Understand that an equation such as y = 3x + 5 is a prescription for determining a second number when a first number is given." [3]. The question shown in Figure 7 was taken from the third quiz in the programming curriculum, and similar to the question in Figure 6, it also requires students to consider an underlying linear relationship. That is, to make a closed polygon in Scratch the character must turn a total of 360 degrees. For a regular polygon with N sides, the character should turn 360÷N degrees between each side. Thus, the number of repetitions needed to create a 10-sided shape must be equal to 10.

This quiz question was considered equivalent to Standard #1.5 because students have to use the given value (*i.e.*, "10") to determine the value of the second number (*i.e.*, the measure of the turning angle). Admittedly, the quiz question in Figure 7 requires additional knowledge not required by the question shown in Figure 6. For example, the second blank (the number of steps to move) is a free variable that determines the size of the 10-sided shape. Also, the linear relationships in this question are tacit, which poses an additional obstacle for students. These additional steps would not be possible if the students had not been introduced to ideas of "Turtle Geometry" [1] and we hypothesize that this represents an advanced application of the mathematical competence described in the CA standard.

Question CSM01966 [3]: Look at the problem below.

□ = △ + 4

If △ = 7, what is □?

A 3
B 7
C 11
D 14

Figure 6. Reproduction of a released question CSM01966 from previous grade four Mathematics test.

Fill in the blanks below to draw a 10 sided shape.

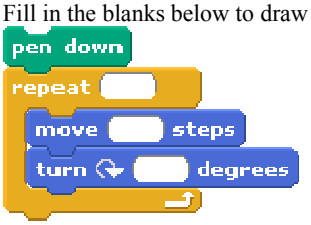

Figure 7. Question taken from the third programming quiz.

5.2 Example 2: Algebra and Functions

Figure 8 shows a reproduction of a released question from the grade four examinations in mathematics. This question assesses a second standard within the area "Algebra and Functions", which is described as "Use parentheses to indicate which operation to perform first when writing expressions containing more than two terms and different operations." [3]. Similarly, the question shown in Figure 9, taken from the second programming quiz, requires students to perform the correct ordering of operations.

The question in Figure 9 places additional requirements on students in interpreting the sequencing of events, but a possible solution requires a similar mathematical calculation. For example

a correct solution to calculate the final value of tempo could include the following expression: 55 + (3x2) + (5x(-3)) + (4x1)

Question CSM10837 [3]: Anna bought 3 bags of red gumballs and 5 bags of white gumballs. Each bag of gumballs had 7 pieces in it. Which expression could Anna use to find the total number of gumballs she bought?
A (7x3) + 5 =
B (7x5) + 3 =
C 7 x (5+3) =
D 7 x (5x3) =

Figure 8. Reproduction of a released question CSM10837 from previous grade four Mathematics test.

What is the value of tempo after we double click on the script below?

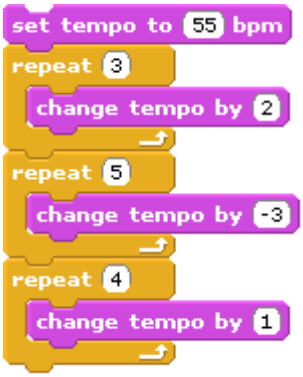

Figure 9. Question taken from the second programming quiz.

5.3 Example 3: Measurement and Geometry

In supporting mathematical understanding, Scratch was particularly effective at promoting understanding of geometric relationships. One of the standards within the area "Measurement and Geometry" states: "Know the definitions of a right angle, an acute angle, and an obtuse angle. Understand that 90°, 180°, 270°, and 360° are associated, respectively with 1/4, 1/2, 3/4, and full turns." [5]. The only released question regarding this standard tested the definition of obtuse angle. However, multiple programming quiz questions directly related to students' understanding of the angles 90°, 180°, 270°, and 360°.

Figure 10 shows a question that requires the student to trace the motion of a character in Scratch. The character begins facing right, which is represented in Scratch as the direction 90°. The character then traces the shape of a square and then re-traces two of the sides of the squares.

The character starts at the arrow, facing the same direction as the arrow. Draw what the character draws when you double click the script. Draw the final position and direction of the character with an arrow.

Figure 10. Question taken from the fourth programming quiz.

6. THREATS TO VALIDITY

In addition to threats to validity identified by the testing agency [4], only 47% of students that agreed to participate in the study submitted CA test scores from 2010. To investigate if there was bias introduced between students that did and did not submit test scores, we compared the average quiz performance between the populations using a Mann-Whitney test and found no significant difference on performance on in-class quizzes between the population that did and did not submit test CA state test scores ($z = 1.255$, $p = 0.210$). Figure 11 shows the distribution of average quiz performance between the two populations.

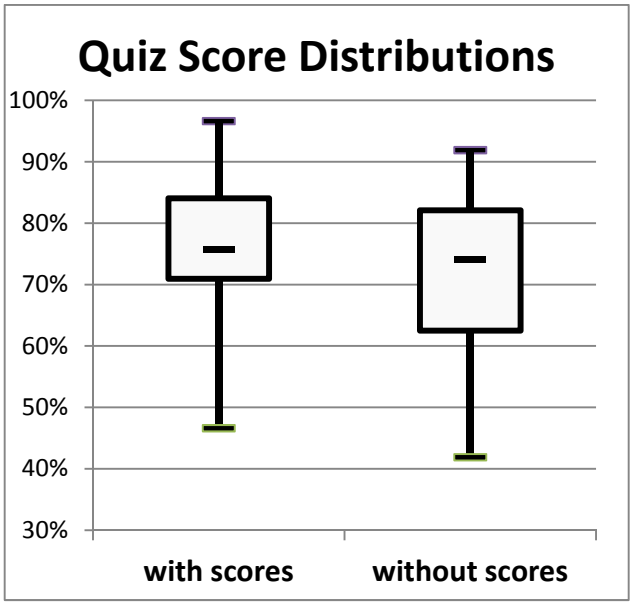

Figure 11. Box plot of average performance on quizzes between students that did and did not submit state test scores.

This paper proposes the hypothesis that the overlap in content between the fourth grade standards and the programming content accounts for the correlations observed. It is possible that these correlations existed within the study for another reason. For example, perhaps CA state exam scores correlate with students' learning efficacy and learning efficacy alone accounted for the difference in performance within the population.

7. CONCLUSION

The current study contributes to the line of research looking for non-programming predictors of performance in programming courses by investigating correlations within a middle-school programming course for the first time. However, the primary contribution is articulating the ways in which mathematics content in the early grades can be built upon and reinforced or enriched through programming curriculum.

While we did not find students' English-Language Arts scores to be correlated with students' quiz performance, there is reason to believe that the students' programming competence would be mediated by their competence assessed by the English-Language Arts test. Students frequently engaged in discussion regarding complex Scratch programs and read information in the online curriculum used in the course. The Grade Four English-Language Arts Standards state: "Students read and understand grade-level-appropriate material. They draw upon a variety of comprehension strategies as needed (*e.g.*, generating and responding to essential

questions, making predictions, comparing information from several sources)." [6].

Researchers have found that prior academic performance, for example in mathematics, is not correlated with performance in an introductory programming course [15]. We hypothesize that this underestimates the extent to which students can and should use content knowledge from mathematics when learning to program. The examples shown here indicate that mathematics that is relevant to students in grade four has remarkable overlap with content from a middle-school programming curriculum in Scratch.

While the programming curriculum was designed to challenge students in learning the programming content, it appears that there was significant reliance on mathematical content knowledge. It is possible that limitations in students' mathematical competence constrained their access to the programming content. While we hope to help all students be successful learning the programming content, there may be realistic constraints imposed by the mathematical nature of computer programming. Ultimately we are encouraged by the opportunity to reinforce and enrich students' understanding of the mathematical content through the programming curriculum.

8. ACKNOWLEDGMENTS

We would like to thank the students and staff of the summer enrichment program. In particular we appreciate the hard work of our undergraduate teaching assistant, Ian Ornstein, who contributed substantially to the teaching and organization of the course. The research reported here was supported in part by a grant from the National Science Foundation (Award #1044106) and through the Institute of Education Sciences pre-doctoral training grant R305B090026 to the University of California, Berkeley. The opinions expressed are those of the authors and do not represent views of the National Science Foundation, the Institute of Education Sciences, or the U.S. Department of Education.

9. REFERENCES

[1] Abelson, H. & diSessa, A. A. (1980) Turtle Geometry: The Computer as a Medium for Exploring Mathematics. MIT Press, Cambridge, MA.

[2] Butcher D. F. and Muth W. A. (1985). Predicting performance in an introductory computer science course. *Communications of the ACM*, 28 (3) 263-268.

[3] California Department of Education Assessment and Accountability Division. (2009). California Standards Test - Released Test Questions - Math Grade 4 http://www.cde.ca.gov/ta/tg/sr/documents/cstrtqmath4.pdf

[4] California Department of Education Assessment and Accountability Division. (2011). California Standards Tests Technical Report Spring 2010 Administration. http://www.cde.ca.gov/ta/tg/sr/documents/csttechrpt2010.pdf

[5] California State Board of Education (1997). Mathematics Content Standards for CA Public Schools

[6] California State Board of Education (1997). English-Language Arts Content Standards for California Public Schools - Kindergarten Through Grade Twelve.

[7] diSessa, A. (2000). Changing minds: Computers, learning, and literacy. Cambridge, MA: MIT Press.

[8] Harel, I., & Papert, S. (1992). Software design as a learning environment. In D. P. Balestri, S. C. Ehrmann, & D. L. Ferguson (eds.) Learning to design, designing to learn: using technology to transform the curriculum. 35-70. Taylor and Francis. Washington DC.

[9] Harvey, B., & Mönig, J. (2010). Bringing 'No Ceiling' to Scratch: Can one language serve kids and computer scientists? Constructionism, 2010, 1–10.

[10] Noss, R (1986). Constructing a conceptual framework for elementary algebra through Logo programming. Educational Studies in Mathematics, 17(4), 335-357.

[11] Resnick, M. (1994). Turtles, Termites and Traffic Jams: Explorations in Massively Parallel Microworlds, MIT Press, Cambridge, MA.

[12] Robins, A. (2010). Learning edge momentum: a new account of outcomes in CS1. *Computer Science Education,* 20(1), 37-71.

[13] Rountree, N., Rountree, J., Robins, A., Hannah, R., (2002) Interacting Factors that Predict Success and Failure in a CS1 Course. *Inroads*, 34, 101–104.

[14] Shanzer, E. (N.D.) Bootstrap. www.bootstrapworld.org

[15] Simon, Fincher, S., Robins, A., Baker, B., Box, I., Cutts, Q., de Raadt, M., Haden, P., Hamer, J., Hamilton, M., Lister, R., Petre, M., Sutton, K., Tolhurst, D. & Tutty, J. (2006), Predictors of success in a first programming course, in *Proc. Eighth Australasian Computing Education Conference* (ACE2006), ACS, pp. 189-196.

[16] Simpson, D. (1973). Psychological testing in computing staff selection: a bibliography. *SIGCPR Computer Personnel Research*, 4, 1-2, 1973, 2-5.

[17] Sutherland, R. (1989). Providing a computer based framework for algebraic thinking. Educational Studies in Mathematics, 20 (3), 317-344.

[18] Wing, J. M. (2006) Computational thinking. *Communications of the ACM*, *49(3).* 33-35.

[19] Yoo, D., Schanzer, E., Krishnamurthi, S., & Fisler, K. (2011) WeScheme: The Browser is Your Programming Environment. *ITiCSE 2011*. Darmstadt, Germany.

[20] Lewis, C. M. (2010). How programming environment shapes perception, learning and goals: Logo vs. Scratch, *ACM SIGCSE Bulletin. 41(1)*, 346-350.

[21] Lewis, C. M. (2011) Integrating Students' Prior Knowledge into Pedagogy. *Proceedings of the International Computer Science Education Research Workshop*. Providence, RI. 139-140.

[22] Lewis, C. M. (2011). Is pair programming more effective than other forms of collaboration for young students? *Computer Science Education. 21(2).* 105-134.