Programming as a teaching tool in third-grade mathematics

Eric Bagley

July 2020

Introduction

Transitions move us from where we are to what comes next. They can be seamless or more difficult depending on the situation. Elementary school is full of those transitions. Some students maneuver through these transitions easily while other students find them much more difficult. In the latter situation teachers often need to be creative in order to find ways to help facilitate these transitions.

Learning the basics of mathematics plays a key role in developing logical and abstract thinking skills and problem-solving skills. For some students, mathematical concepts are simple; they just make sense. For others, these concepts don't come easily at all. These preconceptions often originate from a number of reasons but that is a conversation outside the focus in this paper. The fact is, understanding the complexities of mathematics is difficult for many students to understand.

One of the most challenging parts of education is determining what to teach and how to teach it. Some recent attempts at education reform include the No child left behind Act, the Every Student Succeeds Act and the Common Core Standards. Many education reforms, or at least parts of them, introduce positive and promising results while others perform less than optimal. Regardless of what system is in place, there is a need for teaching tools that help students develop skills that will help them in all aspects of life. Creativity, problem-solving, critical thinking, and communication are vital for someone to become well rounded, responsible individuals that contribute to their home-life, work, and communities.

The focus of the proposed study is not to propose a system that outlines education reform nor to argue that there is a need for it. Its focus is much more

specific: third-grade mathematics. Third-grade is a common time for students to begin learning about multiplication and division, fractions, structures of arrays and area, and describing and analyzing two-dimensional shapes, place value, patterns, and problem-solving, number operations, and measurement and data analysis. The key principles behind learning these concepts are to make sense of problems and attempt to solve them, reason abstractly and quantitatively, construct viable arguments and critique the reasoning of others, model with mathematics, and to look for patterns.

By age 8, children start valuing relationships more, think about the future, and become more independent from parents and family. They can more accurately describe their feelings and understand more about their place in the world. They are beginning to see and understand things from another person's point of view. Their ability to solve problems increases and their attention span gets longer. In math at school, they should be pretty comfortable with addition and subtraction, and they are becoming more proficient with multiplication.

Technology is increasingly becoming a more integral part of our daily lives. One might argue that teaching multiplication and division are no longer necessary because technology can perform those operations for us. Although the latter statement may be true students benefit from skills developed while learning those mathematical operations. Teaching these topics, or at least the principles behind them, are ever more important because technology often does much of the thinking for us. Integrating technology and computational thinking will be extremely valuable to students in their current and future everyday life.

One of the major transitions from second-grade to third-grade mathematics is moving from array-based thinking, using addition, subtracted, repeated addition, and repeated subtraction, to a unit-based, more abstract, thinking, using multiplication and division. This later leads to fractions and ratios. This transition is often difficult for students and a non-traditional approach may prove beneficial.

Another major goal of mathematics at this age is to help the students understand the fundamental pieces of numerical operations and how they are related to one another. Although quick answer algorithms exist, without a strong understanding of the functions behind them it is difficult for students to truly understand the concept. If this is the case, students often just plug in the numbers and hope that they turn out. This may provide correct solutions, but it doesn't lend itself well to solving problems that haven't already been solved. This is detrimental to subsequent learning because every step builds on the previous. If this happens, later on, teachers find that they need to fill in the gaps. This could introduce remedial coursework that may not be needed by other students and may

cause disinterest or lost opportunities for deeper learning.

Understanding the aforementioned misconceptions and shortcomings offer us key insights that can help us develop different teaching tools that help students fully understand fundamental concepts. We propose using programming as a method for teaching multiplication and division at a third-grade level as it could provide an appropriate transition from the group based thinking students have developed from second-grade to the unit based, more abstract thinking they need to develop for future problem-solving skills. It has the potential to assist students in understanding fundamental concepts and fill learning gaps before they become detrimental to both the individual student's learning and the learning of their classmates. It offers an appropriately tailored transition using programming to help them solidify the basics and move to solve more advanced topics.

This study will have a control group and a test group involving two parts, quantitative analysis, and qualitative analysis. The qualitative portion is built off a pre-test and post-test. The qualitative analysis will consist of various pieces: recordings of thinking out loud, pre- and post-interviews, and observation of classroom instruction and participation.

1 Literature Review

Using programming as a teaching method for topics outside of computer science is not novel. There have been many studies related to pedagogy and programming across all educational levels. Within each level, a wide range of topics are covered. Computer Science principles are used, however, to facilitate the learning process. These principles include computational thinking, problem-solving, planning, and creativity. The aforementioned principles have been used to teach language, writing, mathematics, physics, and other science topics.

The studies reviewed can be placed into three categories: ones that teach computer science principles, ones that teach programming languages, and those that use computer science principles and programming languages to teach subjects outside of computer science.

The first area addressed in a number of studies have research objectives that focus on determining appropriate ways to teach a specific computer science principle. Other studies focus on determining appropriate ways to teach programming syntax such as Phanon [EDWARDS]. Based on the target audience there is a variety of teaching software and a variety of different languages that aim to teach different principles. For example, block-based visual programming targets

younger or more novel students, mixed block-text based programming languages such as Droplet can be used for older or familiar students, and text based languages are more easily understood by secondary and post-secondary students. Each of these have varying levels of scaffolding. Block programs heavily utilize scaffolding by providing hints and guiding programmers develop code using close to natural language while text based languages can have little to no scaffolding.

Some programs aim to provide dynamic scaffolding that offers more intensive supports initially but gradually removes some or all scaffolding support for more advanced users.

Learning through Code

The second area addressed in a number of studies is using software programs or programming languages as tools to teach educational subjects outside of computer science. Different than virtual manipulatives or other computer games, programming has proved to be beneficial in helping students develop problem-solving, planning, meta-cognition and other skills [1, 4].

2 Thesis Statement

Idea

Technology is an integral piece of our lives. Digital literacy is vital in everyday life. I am a father; I want my children to have a great education. I have friends who have struggled with math their whole life and I have friends that don't want anything to do with code or math. I think this is detrimental to them because they don't understand that piece of the world around them. I want to help integrate computational thinking into everyday curriculum. I want to determine for myself if programming can be a viable tool for teaching.

Research Questions

1. Does programming math provide better transitions for third graders?

   I think that it could. As one paper said, programming isn't difficult it just requires clear thinking.

2. Is there a better way to teach math (using programming or the more traditional method)?

   I believe that there needs to be deeper understanding. From my conversations with educators it seems that the goal is the same. There have been education reforms with hopes to improve it for all students from primary to post secondary. There is a strong desire to improve but there are some road blocks.

   Those road blocks, I believe, are time limitations; fulfilling a number of requirements imposed by federal, state, and local governments; diverse class sizes, capabilities, and students; lack of funding; Many of these roadblocks can be remedied, while others require more thorough investigation, others may not be feasible to change. Lack of funding is difficult but it can be worked around in creative ways. Divers classrooms can be both beneficial and challenging, providing this type of learning activity may be great for some kids but more challenging for others. Time limitations are difficult to work around but again with creativity hopefully we can develop more effective teaching tools to decrease the amount of time required to teach students, decrease the amount of time it takes for students to learn the concepts, and decrease the amount of time teachers need to put toward reviewing and evaluating performance.

3. Can programming be used for more than creating games and telling stories at this age?

   I believe it can. Using appropriate levels of scaffolding, programming can be a useful tool to help in specific areas of teaching. As Sherin suggests

   > The point here is that programming forces a student to engage with these issues in a way that algebra-physics does not. … it is not too surprising that algebra-physics instruction leaves some undiscovered intuitive cobwebs in this territory, and that the act of programing tends to expose these cobwebs p48 [3]
   >
   > ...programming physics draws more strongly on causal intuitions p50 [3]
   >
   > ...programs might be easier to understand or interpret than equations. 3 [3]

> ...the act of programming requires the students to explode each instant of the motion into a series of actions that happen through what I will refer to as 'pseudo-time' 34 [3]

4. How can we better incorporate digital literacy into our education?

   Using technology in the classroom helps students become digitally literate, but being able to tell a computer what to do increases said literacy.

5. Determine if visual programming can be used to teach third grade students concepts in mathematics.

6. Does use of visual programming assist students in transitional learning?

7. In what ways does programming, and observing the execution of said program help students understand mathematical concepts?

3 Methods

Study

We hope to carry out the study during spring semester of 2021. The study will take place over the course of a few weeks In an in person classroom setting there could be many ways to organize this study. One - Teacher lead: The teacher can lead the students in the programming activity. This may be helpful first starting out to get the general idea of what the programming environment is like and students can ask questions. Two - Student Lead, Teacher Oversee: The teacher can select students to lead the class to solve a certain problem or complete a step. After the problem or step is complete another student can control the computer. Students not controlling can work together to help the controller to write the code.(similar to ladybug programming in [2]). In an online format with a single user, the program would look similar to the in person version. However, in this case, as student would carry out the exercises by her/himself. Both individual and classroom exercises could provide different learning focuses.

In a classroom setting children may give different facial or verbal queues that indicate the level of their understanding. The teacher could also help students at varying levels of achievement succeed by controlling the activities assigned and which solely activities are observed by each student. It may provide students the

opportunity to learn from the experiences of their classmates. A disadvantage of classroom group activities is that each student may not reflect or communicate their actual understanding. This may lead the teacher to think they are either understanding more than they actually are, or that they are understanding less than they display. This may also diminish the motivation of some students as they may not feel comfortable in group settings.

In an individual setting one advantage is that the activity can be tailored to the individual student. This may be more beneficial for lower performing students as it limits distractions, embarrassing moments in front of their classmates, and allows them to perform on their own timeline. Observing in this situation may be more difficult and the data could be more limited than in the in person setting.

A small amount of funds will be dedicated to the students, teachers, and researchers, participating in the study.

Recognizing that teachers have limited time to cover materials, one goal of the study is to limit the amount of extra time needed for these lessons by making them easy to incorporate in their current curriculum. This will involve consistent and thorough communication with the teachers as to the learning objectives, general time dedicated to such objectives, and evaluation periods. The programming environment should not only have a simple interface that is familiar to students, but also provide appropriate levels of transparency and visibility into students' performance that help educators make appropriate evaluation decisions.

We are assuming that all of the students will have some access to a computer and the internet. This may be on a mobile device (preferably a tablet) or a computer with or without a touch screen.

Data

1. Timing data: how long does it take each student to complete an activity.
2. How many of the tutorials, hints, and other help and scaffolding methods were used by the student.
3. Mouse Movements or touch points (possibly, this could tell us what their program usage looks like.)
4. How many blocks did they use?
5. How many blocks did they drag on?

6. How many blocks did they throw away?
7. What did their code look like?
8. What was the running time of their code?
9. Information about there devices can also useful as research shows that touch screen devices provide richer learning experiences.
10. Basic demographic data (ethnicity, household income, do both parents work in or outside of the home)
11. Previous education performance of students (this could just be the pre-test).
12. Pre-evaluation.
13. Post evaluation.
14. Verbal feedback during learning activities
15. Touch or mouse feedback
16. Questionnaire (about the activity: What did you learn? Was it fun? Why? Do you understand [INSERT MATH CONCEPT] more than before? Explain what [INSERT MATH CONCEPT] means? etc)

How to get the data: Pre-test, Post-test, Thinking out loud / ethnography: Record the students using the program: observe their interaction with the program (finger movements, facial expressions, verbal interactions, interactions with classmates)

Data Analysis

Improvement in pre and post-test T-test We don't expect statistical significance (Why?) the study is too small Qualitative assessment (what methods we will use, interview, thinking out loud, etc)

Participants

Students, Teachers, Researchers, Student's Parents (possibly) The students will participate in the programming activities. They may carry them out individually, as a class, or in small groups. They will provide feedback about the programming environment, learning activity, and their general experience.

Teachers will introduce the course material and the learning activities. They may also introduce the programming environment. They help observe the students and their progress. They may provide feedback about the study, the programming environment, the learning activities, and their general experience. The teacher works with the researchers to develop the programming environment and learning activities for the students.

Researchers may provide feedback about the study. With advising from teachers, they prepare the programming environment and learning activities. They also analyze the data and report findings.

Target audience Students, educators, computer science community

Programming Environment

Blockly is developed by Google. It is a platform that allows developers to create apps that use block-based programming. It uses the blocks to create syntactically correct code which can then be used in the application. It can be exported to Javascript, Python, and PHP among others. Developers create the apps "blocks" and then build the app that uses those blocks of code.

It can provide exportable code, it is open source open-source, extensible, and highly capable. For our purposes, we can use this framework to create blocks to be used in the programming environment. The rest of the programming environment would be a web application.

Other programming languages that were considered include the following: Alice, Scratch Blocks, Droplet Editor, snap, Scratch, ScratchJr.

All of these have their advantages, however, they do not provide the same benefits as Blockly. Blockly provides the basic structure for building blocks, and the interface for visual programming. It is built to be only a piece of an application. This allows us to customize and tailor the program specifically to our needs. Some of the other languages and programs considered are more complex than are needed for the purposes of this study. None of them, as far as I have researched, can interface with custom programs as Blockly can.

All in all the main benefits of Blockly include:

1. Provides the library and interface for a visual programming language.
2. It is meant to be a piece of a larger, custom application.
3. It allows for a simple, clean user interface.
4. It, in itself is customizable.

Although it has many benefits, there also comes a cost. As mentioned it is meant to be a piece of the larger picture. That means there is a possibility that building the application around it could be more intensive than anticipated. If teachers have also used other languages before there may be quite a learning curve (which we hope to flatten through a carefully designed user interfaces).

I anticipate that a large portion of my research and thesis coursework will be dedicated to developing the programming environment. Appropriate time estimations would be more accurate after more design work has been completed.

Design of language A scaffolded visual programming language.

Guided learning Transitions through "levels" with Increasing complexity Follows pedagogical practices of learning objectives, with activities that can be adjusted to support those learning goals.

Programming blocks can be limited (you only get 2 or 10 or you can have as many as you need) This could be useful in providing hints to students.

Adaptive programming blocks that can be customized (show this field, don't show this field) to the learning objectives.

Math Topics

Mathematic topics that could be addressed in this study may include the following gathered from the Utah Mathematics Core Guidelines

Reason with shapes and their attributes

Understanding shapes

Understand that shapes in different categories (for example, rhombuses, rectangles, and others) may share attributes (for example, having four sides), and that the shared attributes can define a larger category (for example, quadrilaterals).

Recognize rhombuses, rectangles, and squares as examples of quadrilaterals, and draw examples of quadrilaterals that do not belong to any of these subcategories. One example of a more advanced programming activity could be to write a program that can draw or generate different shapes. If the students are able to write a program that can create one of the shapes above it is a good indication that they understand the shapes.

Verification and Results

Other Risks and Discussion topics

Inability to find a teacher

If we aren't able to find a teacher that would be willing to hold the study as a part of their class, there may be several alternative approaches.

1. Form an after school or online out of school group that students may elect to attend.

2. Make the programming environment available to the public. Many students and/or their parents would be interested in additional activities that can fortify their understanding. This may require some sort of advertizing or to still work through the schools, but it wouldn't require teacher involvement.

3. Attempt to look outside of the normal school locations. This may require additional approvals, and more extensive traveling but it may still be doable.

4. Select a different grade (preferably one close to the original) if there are teachers that would be willing to welcome the study into their classroom

Each of these alternatives would require some extra work and planning. But, should the preferred plan fall through there are a number of other options.

COVID-19 continues through the spring

COVID-19 does not necessarily pose a risk right now. It may adjust the location of the study from the classroom to student's homes, but the bulk of the study

would remain the same. The shortcomings of this approach is that there may not be as much qualitative data to analyze. The lack of qualitative data could limit the findings, weaken conclusions, or leave gaps and unanswered research questions.

4 Implications and potential outcomes

Students will learn math. Computational thinking in real-world situations

5 Conclusion

References

[1] Pamela Fernstrom Doublas Fuchs, Lynn S. Fuchs. A conservative approach to special education reform: Mainstreaming through transenvironmental programming and curriculum-based measurement. *American Education Research Journal*, 30(1):149–177, 1993.

[2] E. Mavroudi G.Fessakis, E. Gouli. Problem solving by 5-6 years old kindergarten children in a computer programming environment: a case study. *Computers and Education*, 63:87–97, 2013.

[3] Bruce Sherin. A comparison of programming languages and algebraic notation as expressive languages for physics. *International Journal of Computers for Mathematical Learning*, 6(1):1–61, 2001.

[4] Nicholas Zaranis Stamatios Papadakis, Michail Kalogiannakis. Developing fundamental programming concepts and computational thinking with scratchjr in preschool education: a case study. *Mobile Learning and organisation*, 10(3):187–202, 2016.