121st ASEE Annual Conference & Exposition
Indianapolis, IN
June 15-18, 2014

Paper ID #10210

On the Road With Codester: Using An Educational App To Teach Computer Science To Grade 1-6 Students

Ms. Gili Rusak, Siena College

I am currently an undergraduate student at Siena College taking classes in computer science and mathematics.

On the Road with Codester: Using An Educational App To Teach Computer Science To Grade 1-6 Students

Gili Rusak
Siena College
515 Loudon Rd.
Loudonville, NY 12211
rusakgili@gmail.com

Dr. Darren Lim
Siena College
515 Loudon Rd.
Loudonville, NY 12211
dlim@siena.edu

ABSTRACT
In this paper, we describe Codester, an original application that teaches young students the principles of programming. The application was developed for the Android operating system, which allows for the use of tablets and smartphones as our teaching medium, instead of traditional laptops and desktops. Codester is the central teaching tool used in a novel outreach program, known as the Computer Science Caravan (CSC). The CSC consists of members of the Siena college community, who travel to local K-12 schools equipped with tablets and other materials to reach out to students who do not usually have the opportunity to learn about programming or computer science. We share our experiences with two such outreach programs.

1. INTRODUCTION

Despite dramatic changes in technology over the past several years, attracting and educating younger students at the elementary school level in the discipline of computer science remains a challenge. Furthermore, only limited use of the developing tablet and smartphone technology has been made to impact the education of K-12 students in this subject. Our research tackles both of these issues and shows how digital technologies can play a transformational role in education to complement the work of Resnick [7].

Educating children in the area of computer science has been the subject of many published works, starting with [4], the seminal work which describes the *LOGO* language and its ability to enrich children's education without the formal teaching environment required of today's complex languages. This work pioneered a new educational method.
Recently, scholars specifically suggested working with K-12 afterschool programs to instill the basics of computer science skills, which include problem solving, computational thinking, and communication skills outside the main school curriculum. Webb and Rosson [11] suggested a set of "scaffolding learning activities" based on Scratch, each of which focused on particular computational thinking concepts like problem solving and abstraction. They showed that their activities could successfully introduce these concepts to middle school females. Intricate projects constructed with such tools required weeks of learning. Some projects took up to several months of work [7].–Yardi & Buckman [12] created an afterschool program for high school students to promote their computational thinking.

We propose to teach even younger students, elementary school children, these computational thinking concepts.

In this paper we first describe our motivation behind creating an Android app as the central teaching tool of the Computer Science Caravan (CSC). Next we describe the app and its teaching goals. We conclude with a discussion of outcomes of our two CSC outreach programs.

2. MOTIVATION

Young students' lack the mathematical comfort may stunt their growth in problem solving. Our aim is to create a problem solving environment that is accepted by young children and sufficiently separate from traditional stereotypical views of mathematical and computer science education. We want it to be valuable, playful, exploratory, and engaging [8]. We strive to teach decision making, and not if-statements or if-blocks; we aspire to teach iteration, and not "loops." Accordingly, we created Codester as a creative [9] and engaging game. We decided that its medium would be an Android app for several reasons.

In previous Siena outreach efforts, exercises were developed utilizing computer programs like *Finch* Robots and *WeDo Lego Robotics*. Both of these technologies relied on the use of laptops with the proper software installed. Since schools did not have sufficient machines for these activities, the faculty mentor and the accompanying students had to bring their personal laptops to use in the activities. However, these were not enough for a meaningful outreach program. Thus, a portable tablet app would solve this problem [10].

Further, tablets have the perception of being accessible devices and children are familiar with utilizing these new technologies. Excitement from our students in our own outreach programs, described subsequently, proved this assumption to be true. According to Kurkovsky [4], the use of mobile devices is an important part of students' lives, and involving them in a student's learning period helps gravitate them towards computer science and helps engage them in their studies. Additionally, students have recently associated programming with these devices as opposed to computers because they are such an integral part of everyday life [4]. Recent literature supports the use of these devices as a valuable teaching tool for K-12 students for the computer science field [4, 10]. Our program uses the tablets for educational purposes to show students how useful they can be in play and learning alike. Thus, developing a tablet application with the main purpose of the CSC in mind, we were able to focus particularly on what concepts we saw useful in computer science education.

Ultimately, our design decision for the CSC focused on an outreach program that was accessible to younger students. Unlike the other programs, we acquired Android tablets and channeled their technology to allow our outreach program to be engaging, unique, and independent. At a fraction of the price of laptops, the tablets were easily obtainable. Thus, we are able to visit all types of schools and communities, suburban and inner city alike, without being dependent on their computing facilities.

3. CODESTER, THE APP

Computational thinking can be described as solving problems that can be carried out by an "information-processing agent" such as a computer or a tablet in our case [2]. Our app teaches computational thinking.

3.1 Game Play

Codester is a level-based game. Each level has its own puzzle, some with many possible solutions. In each level, the Codester is placed somewhere on the grid with a destination space demarcated with a star symbol. The object of a level is to move the Codester from its starting spot to the star symbol, using directional arrows which act as instructions. The user drags appropriate commands (arrows or symbols), from the command zone to the code prompt in a shape of paw prints, as shown in Figure 1. In effect, the commands form the "programming language" of Codester. A set of commands creates a code prompt. Once every paw print is filled, the play button can be pressed, and Codester will follow the instructions laid out in the paw prints, from left to right. The user will see it move one instruction at a time, so they can follow the movements as Codester "executes" them. If Codester is successfully moved to the star square, then a popup message stating so will appear, and the app will let the user move to the next level. If Codester is unsuccessfully moved (either by moving off the grid or ending at a location other than the destination space), an error message pops up, giving the user two options: starting the level over again, or returning to the last (albeit, unsuccessful) sequence of commands. The latter option allows the user to make edits to the sequence, in order to win the level.

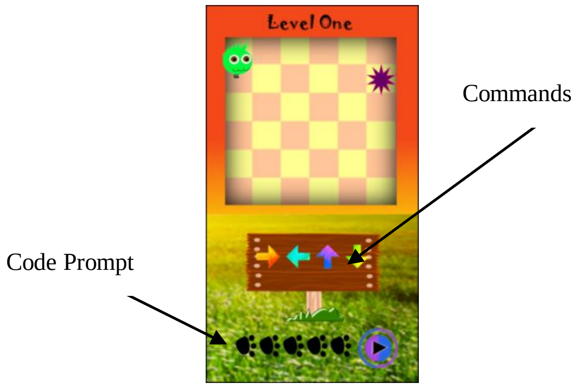


Figure 1: The application's user interface.

3.2 Programming Concepts in Codester

3.2.1 Sequencing

The process of dragging and dropping commands teaches users the most basic principle of computational thinking: sequencing of instructions. Directions must be placed in a specific order, for Codester to obey.

3.2.2 Code Reuse

The process of code reuse, or subprogramming, is the idea that one can create a new function (using existing functions and statements), name it differently, and then be able to use it several times throughout the code. The user has the ability to make a subprogram within the instruction set on some levels of Codester.

This is done by using a “Create New Command” button. On some levels, the user must make a sequence of directional movements that will be represented by a smiley face icon, a new “command”. When finished, the smiley face icon can be dragged into place along with the other direction arrows. The new command can be used multiple times or not at all. Users will have to determine, based on the level, how many times to use the new command. For each level, the user may currently make one new command. A future version of Codester may expand this ability to multiple commands.

For example, if the user had fills those paw prints “down” then “right”, each time he drags the smiley face icon to the code prompt, Codester will execute “down” then “right.” The right picture in figure 3 shows decision making.

3.2.3 Decision Making

In Codester, decision-making is portrayed in the form of a conditional. The user must decide whether or not to utilize the conditional command. At some levels, specially colored squares begin to appear on the grid. These squares come in pairs (the two colors that are used are green and light blue, see figure 3). When the user moves Codester to a square, he may move off the square as normal, or have Codester jump directly to the other square on the grid of that same color. This particular kind of move can be thought of as a teleport, or jump. This is done by dragging a blue (or green) square command from the command zone to a paw print.

3.2.4 Iteration

The concept of iteration is repeating the same function several times. In Codester, iteration is taught by a “repeat three” command. At later levels of the game, a user can drag a “number three” command from the command zone. This three may be applied on top of any command (either direction commands or newly made commands) in order to have that movement be repeated exactly three times.

3.2.5 Design Considerations

We made several important design decisions in Codester; in particular, we chose to make it

- A level-based application
- A gender-neutral application
- Utilize warm colors
- Use accessible, inviting graphics
- Use general symbols that appeal to everyone

4. THE COMPUTER SCIENCE CARAVAN

4.1 Participants

In the fall of 2013, we ran two CSC outreach programs for elementary school students to evaluate the effectiveness of our new tablet methods and Codester. Participants were selected on a first-come-firstserve basis through an after school program at a local suburban elementary school in New York. The workshops were divided into students in grades 1-3 and students in grades 4-6. Prior questioning reflected that almost none of the students had heard of programming, let alone worked with it.

For the 1st-3rd-grade user study, there were 17 participants all with informed consent from parents. For the 4th-6th-grade study, there were 15 participants, 14 of which had informed consent from parents.

Students in our user studies had some exposure to computers and technology at home before the program. All students mentioned that their family had either a laptop or desktop computer at home. Eighty seven percent of students mentioned that their family had a tablet. Most students reflected that they played an average about half an hour on the computer every day. Some mentioned that they played as much as three hours but this was only a couple of students. All students had played either video games or computer games and either games on tablet or games on smartphone.

4.2 Workshop Logistics

Each workshop was a four-session program where students experienced and learned about programming.

Because students were novices and had no experience with programming, the first session was an offline introduction to programming. This helped gauge the students' knowledge before the program. Students were introduced the coding as a process of giving a computer directions. They were given the task of writing detailed instructions for building a tower out of blocks. In the 1st-3rd-grade study, students had difficulty with this task since their writing and reading skills were not yet developed. This is a main reason why programs like actual programming languages would be less effective for these age groups, while Codester does work because it doesn't require reading or writing skills, it only requires logic. On the other hand, the 4-6 grade students were very excited about this activity and each group gave detailed instruction for different, clever tower patterns from the same blocks.

Additionally, during the introductory activity Codester was introduced on a board that paralleled the game. Students learned how to write code for Codester on the board by drawing arrows and symbols on it, mimicking the game-play, see figure 2. This method proved to be very helpful in explaining programming and Codester to the students.

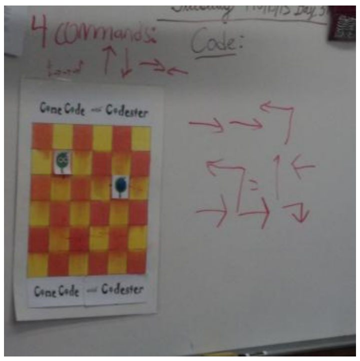


Figure 2: Codester code on the board.

During the remaining three sessions, students developed their computational thinking skills, described in section 3, by playing Codester on tablets that we provided for them. Two or three students shared a tablet so that they could learn from one another and collaborate on solutions to the levels. All students had to solve every level before moving to the next level to ensure that everyone understood all of the concepts. Students were encouraged to find different solutions to the same level; a vital skill emphasized by Resnick [7]. Students were able to progress through about three-five levels per session and, by the end of both workshops, students were introduced to all of the computational thinking concepts that Codester offers.

Surveys and personal interviews were conducted throughout the sessions to determine students' understanding and awareness of computers and computational thinking. Students took pretests and posttests in each workshop to gauge their increased knowledge throughout the sessions. Although both the 1st-3rd-grade and the 4th-6th-grade tests checked the same computational thinking concepts, each was adjusted for age appropriate problems.

These observations and interviews were essential to help us understand the way the participants developed and what they really gained from our workshop, like the methods of [1].

5. Results

5.1 Preliminary Knowledge

The first session in each group led an interactive discussion to introduce and familiarize students about programming; what it is, what it involves, and other information about computers.

Students showed diverse knowledge in answering specific questions.

"What is coding?" students were asked during the first session.

Grade 1-3:

- *Typing code*
- *How to make things happen*
- *Doing math*

Grades 4-6:

- *Kind of like programming*
- *Passcodes, secret messages*
- *It's like making an app.*

"What are the differences between humans and computers?"

Grades 1-3:

- *Computers can't talk to each other*
- *Computers are smarter*
- *Computers are electronic and not living*
- *Computers don't have a brain; computer doesn't know what to do.*

Grades 4-6:

- *Computers have more information than humans*
- *Computers don't have a brain*

- Computers need to be told what to do

These preliminary interviews reflected that several students had good ideas about how computers worked and what programming was. However, only about 15% of students were able to contribute to these answers on the first session.

By the second session, after using Codester and coding for the first time, about 50% of the students from both workshops were able to contribute to these questions. They learned the "arrow and symbols" programming language of Codester, and this clarified to them certain important aspects of programming. Students mentioned details like:

- Computers need to be programmed in a special language

- We need to give the computers directions

- [Directions] need to be specific and simple

- Computers don't have logic.

Specific to Codester, students reported that they learned:

-The arrows will move Codester to the target.

-The arrows make up the directions for Codester.

-A command is telling what to do being specific.

5.2 Learning CT through Codester

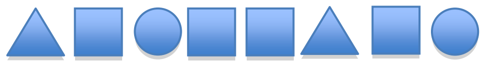
5.2.1 Sequencing

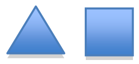
Through Codester, students progressed through the first levels easily showing their understanding of sequencing, the first computational thinking concept that Codester teaches.

5.2.2 Code Reuse

The second concept of code reuse and creating a new command was first presented on the board to the students. They were given a problem like:

"Please look at the following pattern.



We want to write a ☺ instead of 

How will our pattern look now?"

Although only 18% of 1st-3rd-grade students and 50% of 4th-6th-grade students were able to answer this question correctly on paper at the beginning of the sessions, when presented this concept through the tablets, the concept was clarified to the students and they were able to proceed through the levels that required it. By the end of the program, 50% of 1st-3rd-grade students and 93% of 4th-6th-grade students were able to answer the above-mentioned question correctly, showing that they understood this concept.

Students further understood why the concept was important.

-It is kind of the same thing but it takes less spaces

-It shortens the code.

The aspect that presented them with the most difficulty in regards to code reuse was having to use the same new command that they created more than once. However, all students, with some help, eventually grasped the concept. This taught efficiency and introduced several ways to solve the same problem.

5.2.3. Decision making and Iteration

Students were able to understand decision-making and iteration through Codester on their own based on the user interface see figure 3.

-We have to decide if we want to use the 'portal'

-It jumps

It is important that the students understood these concepts on their own because it ensures that others who use Codester without direct workshops and instruction will also be able to learn these concepts.

By the last session, all groups from both the 1-3 and 4-6 studies arrived to at least level 9; until where all major computational concepts of the game are introduced. This rapid learning pace compares well with more traditional approaches such as teaching a programming language directly. This pace at which users learned to use the game is unique to our design.

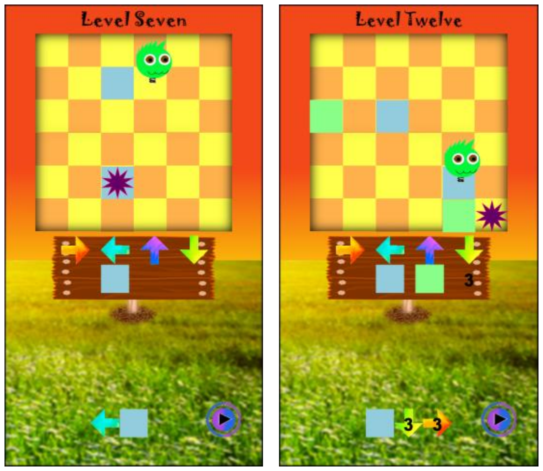


Figure 3: On the right is the decision making interface; on the left is the iteration interface.

5.3 Effectiveness of Codester

Codester was effective in teaching students programming concepts. Questionnaires showed improvements in logic skills and computational skills of students after using Codester, summarized in Figures 4 and 5. Users improved in every area that Codester aims to teach from the beginning to the end of the sessions. The greatest improvement was with the Code Reuse concept, which students were mostly unfamiliar with before the sessions while about half of them understood after using Codester.

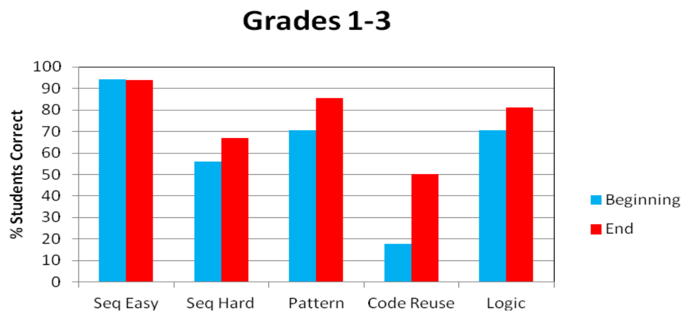


Figure 4: Improvements from the 1st-3rd grade study.

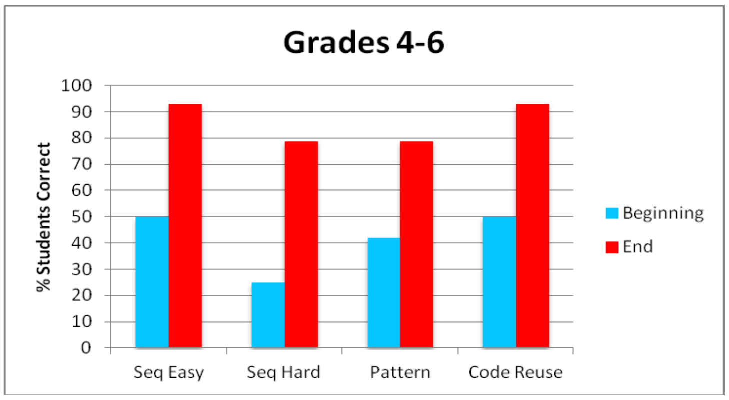


Figure 5: Improvements from the 4th-6th grade study.

5.4 Classroom Engagement

Interviews reflected that students learned to give more detailed instructions with more logic after using Codester. When asked to give directions about building a tower from blocks before using Codester, many students responded with “pick up block and place on top of the other.” After Codester, students responded “take block, lift it, and place it on top of the other block.” This showed how Codester improved the computational concept of giving detailed directions.

Observations showed that students were very enthusiastic about the workshops and the game. Students were passionate and determined to move through the levels. When asked, *“Who wants to use the tablets?”* everyone raised their hands, some even enthusiastically jumped out of their seats saying *“Me!”*. Every session, students came with an engaged, open mind, determined to learn.

In all, 83 percent of students in our user study expressed that they loved using tablets to learn programming concepts. Any user who owns an Android tablet or smartphone is able to play it from anywhere. Because almost 90% of students in our studies owned a tablet, this allows for greater flexibility of Codester. Unlike *Lego Mindstorms*, for example, which requires materials, software, and space, Codester is mobile and convenient.

Users also enjoyed the app, many comparing it to the *Minecraft* app and others. Fifty-eight percent of students (1-3) said that it was challenging but they understood the gameplay, and the rest said that it was easy to use. Seventy-three percent of 4-6 students felt that the game was easy. No student felt that Codester was too challenging to the extent that they could not understand the game at all; :it was accessible to students in grades 1-3 and 4-6 alike. Eighty-two percent of 1-3 students said that they would definitely play the game again. Sixty-seven percent of 4-6 students said that they would like to play it again.

Parents and students alike were enthusiastic about the CSC after school program. On the first day of registration, 32 spots were filled and we were forced to split it into two sessions, since the demand was so high. At the end of the sessions, one parent wrote:

- *"We felt lucky that [our son] got into this club. He was excited to go on Tuesdays and would tell us about it. We thought it was a great way to teach kids computer skills at a young age."*

Another parent wrote: *"it made him realize he's good at things like this- or at least is more interested in these concepts."*

Most students reflected that they were excited about the workshops and some even inquired about future sessions without being asked. Girls particularly mentioned their great enthusiasm for the club. Many parents were interested in future sessions of the program. This proves the mission of the CSC: teach young students about computers and programming and involve them in these subjects in an engaging and meaningful manner.

6. CONCULSIONS AND FUTURE WORK

From the initial stages of the project, we envisioned having a version of Codester available on the Google Play App store for downloads by other teachers, educators, and groups. We wanted it to be a free download, so people could try it out and give concrete suggestions back to us. Codester was submitted and accepted by Google Play on 12/09/2013 with no download cost.

Our experience has shown the effectiveness of new tablet technology on the learning experience of the young K-12 students. The app was engaging and the kids tended to focus and learn new concepts from it. It exhibited promising results and some students inquired about seeing the rest of it to play it again. When young students engage in logical, educational apps and programs like those run by the CSC, they will develop and acquire knowledge for more complicated problems connected to computer science in general. We plan on running additional outreach programs under the CSC in more schools and youth organizations in the near future.

7. REFERENCES

[1] Brennan, K., & Resnick, M. (2012). New Frameworks for Studying and Assessing the Development of Computational Thinking. Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.

[2] Cuny, J., Snyder, L., & Wing, J.M. *Demystifying computational thinking for non-computer scientists*. Unpublished manuscript in progress, 2010. referenced in http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

[3] Goadrich, M., Rogers, M., Smart Smartphone Development: IOS Versus Android. *Proceedings of the 42nd ACM technical Symposium on Computer Science Education*, Mar. 2011

[4] Kurkovsky, S., Engaging students through mobile game development. *Proceedings of the 40th AMC Technical Symposium on Computer Science Education* March 3-7, 2009, 2009.

[5] Papert, S., *Mindstorms: children, computers, and powerful ideas*, New York, NY: Basic Books, 1980.

[6] Reilly, M., Kindergarten coders can program before they can read. *New Scientist* 2927, 21-22, 2013.

[7] Resnick, M. All I Really Need to Know (About Creative Thinking) I Learned (By Studying How Children Learn) in Kindergarten. ACM Creativity & Cognition conference, Washington DC, June 2007.

[8] Resnick, M., & Rosenbaum, E. (2013). Designing for Tinkerability. In Honey, M., & Kanter, D. (eds.), *Design, Make, Play: Growing the Next Generation of STEM Innovators*, pp. 163-181. Routledge.

[9] Rusak G., Lim, D. (2014). In preparation for the Proceedings of 2014 Consortium for Computing Sciences in Colleges — Northeastern Region.

[10] Sharples M., The design of personal mobile technology for lifelong learning. *Comput. Educ*.34, 3-4 (Apr.2000), 177-193, 2000.

[11] Webb, H. and Rosson, M. B. 2013 Using scaffolded examples to teach computational thinking concepts. *Proceedings of the 44th ACM technical symposium on Computer science education* (Denver, CO, March 6-9, 2013) ACM, 95-100, 2013.

[12] Yardi, S. and Bruckman, A. What is computing?: bridging the gap between teenagers' perceptions and graduate students' experiences. *Proceedings of the third international workshop on Computing education research* (Atlanta, HA September 15-16, 2007) ACM, 39-50, 2007.