Fun with Functions

(Using Functional Programming in Middle School Algebra)

Ole Bjelland, Alex Gendreau, Kari Santos, Anna Villani

Abstract

We explored whether or not using a modified version of the Bootstrap[1] curriculum could enhance students mathematical ability and increase their interest and belief in their ability in mathematics and computer science. We visited a remedial middle school mathematics classroom with ten students six times over a three week period to provide instruction and take observations. Unfortunately, we do not have significant statistical results because we did not target the surveys/tests specifically enough but we have several meaningful ethnographically observed stories that will guide future experiments.

Section 1: Introduction

Getting computer science into the K-12 public school system is a challenge of curriculum, resources, and time. An option that has been explored with some success recently is integrating computer science within the mathematics curriculum. Bootstrap World is an organization that utilizes functional programming to find alternative methods for teaching mathematics in middle school. This essay will detail the methodology, observations and reflections of a three-week project, Fun with Functions, implemented at a middle school in Boulder where a revised version of the Bootstrap curriculum was taught. The ever-increasing importance of STEM subjects in the digital economy creates an incentive and responsibility to ensure that students of tomorrow are ready to meet the challenges that lie ahead. Fun with Functions aims to analyze and evaluate the extent to which Bootstrap helps students conceptually understand and become competent in mathematics in addition to discovering if Bootstrap also positively changes one's attitude towards mathematics. Fun with Functions have therefore made two research questions that this study will aim to answer.

Research Questions:

1. Can alternative mathematical instruction help students understand topics that they were previously struggling with? (Measured through pre/post test)
2. Through the bootstrap program, can we increase students interest in mathematics and computer science? Can we increase the belief of the student to be successful in mathematics and/or computer science? (through pre/post survey)

Based on the existing literature, this paper hypothesizes that functional programming (through the use the Bootstrap curriculum) positively increases competence and attitudes towards mathematics.

[1] http://www.bootstrapworld.org

In Section 2, we review the current literature in Bootstrap and ethnographic observations. In Section 3, we discuss on experimental methodology and results. In Section 4, we discuss our results and the lessons we learned from our experiment and in Section 5 we conclude and propose lines of future work.

Section 2: Literature Review

2.1 Bootstrap and CS in the Mathematics Classroom

In today's education system, it is common for children to reach a certain point in their education where they decide that they are not good at mathematics. Mathematics is generally seen as a process. When the process gets so complicated for students that they become continuously frustrated, this will likely push them to the belief that mathematics is just not for them. Some claim that using functional programming to teach mathematics could be a way to solve some of these issues**.**

In United States primary and secondary education, the three most valued aspects are reading, writing, and arithmetic. It is not uncommon that any new material that competes with these core elements is marginalized. Felleisen claims that a new aspect needs to be added to that list: programming, and that a full-fledged programming language can, in fact, be used as a form of mathematics (Felleisen 2009). This could lead to a more natural progression of mathematical concepts; as students code, they discover concepts on their own because they want to add more features to their programs. One problem with this is that if the notion of "function" in programming differs from the notion of "function" in mathematics, it can cause a mental block for students. A language created that is in harmony with mathematics, involves algebraic manipulation of images and data, and minimal overhead in the Integrated Development Environment (IDE) would be a great medium for students to explore and engage in mathematics.

Wright claims that functional programming is a useful way to both introduce students to programming and present algebraic concepts. Imperative programming uses constructs such as loops, mutable variables, and procedures. Since these constructs are not related to algebra, students learning an imperative language would learn syntax and programming structures rather than algebraic concepts. Wright claims that are not enough studies to back the claim that teaching Scheme helps students with algebraic skills, instead the idea is presented as a natural consequence of the functional nature of Scheme (Wright, Rich, Lee 2013). Thus, there is a need for more investigation on the effects of learning mathematics through functional programming using Scheme. Another big claim Wright makes in favor of functional programming is that it allows students to interact with mathematics in a way that "brings it alive". This way, they can gain a deeper understanding of the power of mathematics.

Bootstrap is a computing curriculum designed to reinforce specific learning objectives in algebra. Bootstrap's programming model and choice of a functional programming language

(Scheme) allow students to directly apply problem-solving processes for programming to standard problems in algebra. Schanzer hypothesizes that these design strategies enable transfer (Schanzer, 2015). Transfer of learning between domains typically requires both deep structural connections between the domains and explicit instruction in how to apply concepts from one discipline to the other. This raises two challenges: can we identify specific problem-solving practices in computing that have direct analogs to processes in mathematics, and can we teach them in such a way that students realize performance gains in mathematics?

2.2 Literature Review of Pedagogical Methodology

Hulleman et al suggest that “Things indifferent or even repulsive in themselves often become of interest because of assuming relationships and connections of which we were previously unaware” (Hulleman et al, 2010). To give an example, many a student has found learning mathematical theory difficult or unattractive but when studying engineering and having to utilize this theory as a necessary tool, Hulleman et al suggest the attitude towards the mathematical theory may positively improve (Hulleman et al, 2010). The Fun with Functions project aims to positively improve attitudes towards mathematical theory by engaging students with functional programming through the Bootstrap curriculum.

Additionally, “positive expectancies or a sense of competence can enable individuals to perceive value in activities.” (Hulleman) Hence the Fun with Functions project aims to empower the students through functional programming to increase perceived value in mathematical theory which could “increase task engagement and the development of competence and positive performance expectations (Eccles & Harold, 1991; Eccles & Wigfield, 1995).”

However a significant body of research suggests that perceived task value is correlated with interest and achievement choice rather than performance (e.g. grades) which is more closely correlated with subjective expectancies instead (Eccles & Harold, 1991; Updegraff, Eccles, Barber, & O’Brien, 1996; Wigfield, 1994; Xiang, Chen, & Bruene, 2005; see Wigfield & Eccles, 1992, for a review). In terms of pedagogical methodology one must therefore account for the possibility that telling students who perform poorly in mathematics or do not find it interesting, that mathematics is important for the future may be received as threatening and intensify negative reactions. The Fun with Functions project therefore only demonstrates the importance of mathematics indirectly through having the students utilize mathematics as a tool when conducting functional programming tasks.

Furthermore, Hulleman et al suggest that a way to further prevent these negative reactions is to encourage students to internalize the relevance and importance of the course material to their own lives by generating their own connections and conceptual discoveries. The Fun with Functions project will support this premise by assigning tasks that engage personal traits or stories for each student.

2.3 Noticing Classroom Interactions

There is much research on how to improve mathematical education in K-12. In the classroom teachers are expected to adapt their instruction based on ideas that students raise (National Council of Teachers of Mathematics, 2000). An important aspect of this adaption is how teachers “notice and interpret classroom interactions”(Van Es and Sherin, 2002). Noticing classroom interactions is complex - teachers need to determine what interactions are important to the situation and connect that to broader concepts of teaching, and they need to tie in their knowledge of particular students and subject matter in order to make sense of these situations. Only after noticing, can the teacher adapt. (Sheri and Van Es, 2005).

Section 3: Experimental Research Methodology and Results

Our research relied on gathering both qualitative and quantitative data to discover if and how Bootstrap positively affects students’ understanding and attitude towards mathematics and computer science. The quantitative data would be used to measure whether Bootstrap actually enhanced the students’ understanding of mathematics while the qualitative data would be used to measure whether Bootstrap enhanced students’ attitude towards mathematics and computer science.

The quantitative data was collected using a pre and post test that tested the students’ ability of mathematical concepts that are related to what the Bootstrap curriculum teaches. The qualitative data was gathered using two forms of ethnographic research methodology: participatory observation and silent observation. As Emerson suggests, “Ethnography is an active enterprise. Its activity incorporates dual impulses. On the one hand, the ethnographer must make her way into new worlds and new relationships. On the other hand, she must learn how to represent in written form what she has come to see and understand as the result of these experiences.”[2] An ethnographic field study demands a study of your prospects, accurate context-specific recording of these studies and a introspective consciousness of one’s own biases in these recordings. Our team decided to assign four different roles to better manage the dynamic reality of conducting research while simultaneously teaching the class. The four different roles are: Facilitator, Teaching Assistant, Participatory Observer and Silent Observer.

A. Facilitator

The classes were predominantly a mix of a lecture session followed by a form of semi-structured exploratory learning session where students could apply what they had learned towards building what they themselves wanted within the framework we were working (e.g. shapes). The facilitator led each session by conducting the lecture part of the lesson as well as facilitating individual learning needs during the semi-structured exploratory learning session.

B. Teaching Assistant (TA)

The TA addressed ongoing individual students’ problems both with conceptualization of lecture material as well as the collaborative and interactive tasks the students worked on during class and any technical problems the students ran into.

---

[2] Emerson, R. M., ‘Writing Ethnographic Fieldnotes’, University of Chicago Press, Chicago, 1995

C. Participatory Observer
The Participatory Observer was one of the most essential roles on the team as they focused solely on observing and recording their observations while being able to engage with the students. We estimated that Middle Schoolers are a relatively silent (non outspoken) group so it was very helpful to have an observer who could actively engage with the students to try and get more holistic and contextual answers when conducting the ethnographic observation.

D. Silent Observer
The Silent Observer had two main objectives. The first was to limit potential biases from the Participatory Observer by having an account of observations who did not interact with the students but observed them from a distance. The second objective was to catch the more general dynamics in the classroom with a special focus on mapping the general attitude and motivation of the students towards the learning environment and curriculum at different points in time during the class.

Our team believes this ethnographic methodology effectively created time and energy for different team segments to focus on their respective tasks whilst ensuring a productive class and reliable research.

3.1 Experimental Design
To investigate the first research question, we administered a pre and post test of mathematical ability targeting the concepts we planned to teach the students. Copies of the pre and post test are available in Appendix A. Through the results of the test, we hope to see results similar to other ability measurements performed on the Bootstrap curriculum[3].

To investigate research question number two, we administered an interest/utility value survey based on the survey outlined in (Hulleman et al). The survey measures pre and post interest and utility value on a seven point Likert scale. The survey Hulleman et al gave was targeted at undergraduates who were taught a new multiplication technique. They first solved the problems using the new technique and took pre-survey and then participated in a utility value intervention (an essay about the applicability of mathematics in their lives). After the intervention, the students took the test again and completed the post-survey. While we were not particularly interested in studying utility value at this point in our experiments, we chose this model because it was a paper the team was familiar with and provided sample questions directly applicable to learning new techniques for mathematics, which is one of the goals of Bootstrap. We modified the survey questions to be applicable to the Bootstrap curriculum and to include some questions on computer science. The specific questions we used are available in Appendix B; we used a five point Likert scale because of the age group we were targeting. As in the Hulleman et al paper, we used reversed questions to measure the reliability of our results.

[3] http://www.bootstrapworld.org/impact/

3.1 Experimental Outline

In this section we will outline our plans for our six visits to the classroom. We decided to visit the class six times based on the students' testing schedule, and how much time we wanted to analyze our data. Each class we went to was either 50 minutes or 60 minutes, so we had to shorten each 85-minute Bootstrap lesson into 50 minutes. We allowed the teacher to choose lessons based on curriculum standards that were recently taught or needed more reinforcement.

We conducted our experiment on a 7th grade mathematics class at a middle school in the Boulder Valley School District. The class is an elective, so they have a mathematics class twice a day for extra support. At this school, 40% of the students are on free/reduced lunch. The class consisted of ten students: five boys and five girls. Six students were also English Language Learners. We allowed the teacher to choose lessons based on curriculum standards that were recently taught or needed more reinforcement.

- **Day 1:** Introductions, pre-test and pre-survey, Bootstrap Hour of Code
  During the first visit to the classroom, we introduced ourselves of to the students and explained what we were going to be doing with them over the next few days. We administered the pre-test and pre-survey

- **Day 2:** During day 2 we taught a revised version of Unit 1 of the Bootstrap curriculum. Unit 1 mainly consists of three different stages. Together these stages introduce computational thinking on how video games are built along with how the different elements on the screen interact. Stage one was to 'dissect a video-game' where students were to identify what characters existed in NinjaCat (a simple two dimensional online game) and how those characters changed in terms of X and Y coordinates when you played the game. Stage 2 was based on conceptualizing functions and their order of operations using circles of evaluation. This stage focused on using numbers in the circle of evaluation while stage 3 incorporated shapes as well, allowing students to produce different shapes on the WeScheme editor by the end of class.

- **Day 3:** Function contracts, error messages, images
  We completed a modified version of the second unit of the Bootstrap curriculum. The focus was on function contracts which discuss the idea of domain and range of functions, which can be a hard concept from students to grasp. We introduced the idea of nested functions (function composition) and tried to make a connection between function contracts and circles of evaluation. Two analogies we used for function composition were puzzle pieces and legos. We explained how to read the error messages associated with the function contracts in the hopes of making the students more independent programmers during the work time portion of the day. During the work time portion of the day, the students we challenged to build unique images using the provided function contracts for shapes (which produce images) and function contracts for changing images (i.e. rotation, scaling, and overlaying). The goal of the

work time was to give the students an opportunity to practice using the concepts they had just learned and to develop connections between the circles of evaluation and the Scheme code.

- **Day 4:** Definitions, Defining Functions
  We taught a version of Unit 3 from Bootstrap. The main idea of this lesson was to introduce definitions and how they make coding easier(for example, creating an image by combining many separate images). This idea can be applied to the algebraic concept of defining a variable (e.g, x = 5). Just how the value 5 is stored in the variable x, we can create an Image in Scheme, and store it to be used again. We also introduced defining functions and went through an example of how to do that. Then, we went through building an image of the Colorado state flag together. This involved first creating the separate components needed for the flag using definitions (outline, stripe, sun, C), and then putting them together correctly. We had planned to give them some time at the end to explore with making flags, but the lecture went longer than expected. However, it got the students at least thinking about the flag project.

- **Day 5:** Teaching Functions to Compare
  We taught a version of Unit 6 from Bootstrap. The main concept of this lesson was boolean functions. The students had not learned the term boolean before, so some time was spent explaining, that unlike other types (like Numbers) which have an infinite number of values, boolean variables can only be true or false. The students were familiar with the comparing functions like < or =, so we were able to make a connection between these functions and the return value of true or false, stressing that the return was not “right” or “wrong”. This was followed by an activity where the students were given a Bootstrap game program that moved a butterfly around the screen by pressing arrow keys. The goal of the activity was to write functions that would stop the butterfly from leaving the screen. An example function “safe-left” returned true if the x position of the butterfly was a value that would not be off the left side of the screen. The students were prompted to write similar functions for the the other 3 sides of the screen.

- **Day 6:** Flag Project, post-test and post-survey, wrap up
  For our final visit to the class, we wanted the students to have an opportunity to showcase what they had learned. The facilitator began the lesson with a brief introduction to the symbolism of shapes and colors in flags and then the students were tasked with designing and creating their own flag in Scheme. We encouraged the students to spend some time designing [on paper] before beginning to code. After the students had completed their flag projects, we administered the post-test and post-survey. We then did a walk around where all students could see the other projects. Our final visit ended with a circle share session led by the teacher to get some more feedback from the students.

- **Follow-up:** We met with the teacher the week after our experiment concluded to discuss each student in the class in order to have more information in which to group our observations and reflections.

3.2 Individual Stories

In this section, we examine in more detail the experiences of four students in the classroom. We explore both observations during the classes and background information including the discussion with the teacher about each student after our experiment had concluded. Note to protect the students' privacy, all names have been changed.

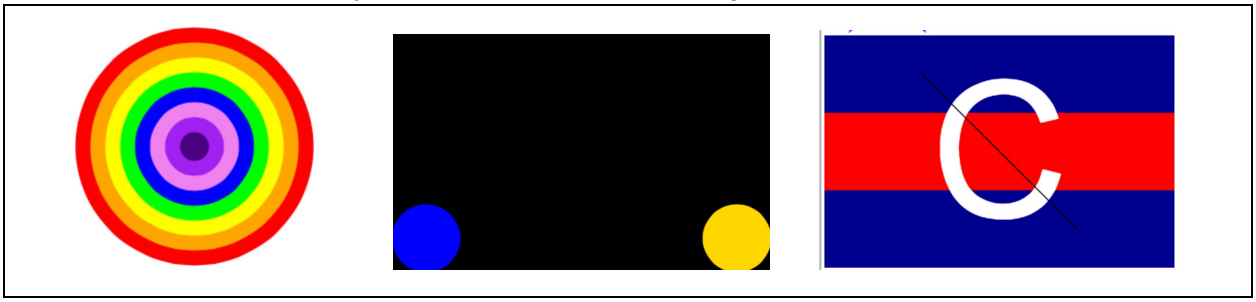


Figure 1: A sample of the images the students produced. From L to R: A target drawn by Todd, a flag by Nicole, and a flag by David.

Todd

During the first two classes Todd was unengaged and frequently off task. He had to be constantly reminded to be working and would only attempt the activities if his teacher or one of us were with him. However during the third class period, when we taught domain, range, and function composition, he became a much more active participant. During the independent work time at the end of the class, he produced the "coolest" image of the class shown in Figure 1. His classmates, teacher, and us all praised him for his work. He seemed extremely happy after the class. During the fourth class period, his participation continued. Todd was struggling with the warm-up (circles of evaluation, but once we helped him work through an example, he caught on quickly. He answered several questions in class today, which was a first. However in the last two classes, Todd and David (an unengaged student we discuss later) sat together again and we think this hurt Todd's engagement and enjoyment although he was still a more active participant than he had been during the first two days of class.

David

David was strongly encouraged to take the extra mathematics elective. It is not because he struggles in mathematics necessarily, but that he does not appreciate school so he does not do the work. He is unmotivated in all of his classes, and comes from a family that does not make school a priority. Our observations of David throughout the course match this description. He did not pay attention during the lectures, and had to be constantly prodded to do the assignment during work time. According to his teacher, he is happy when he can create things. He spends his free time designing logos and t-shirts. For the majority of the experiment, the observations showed a lack of interest. However, there were a couple situations that we observed some

interest in Bootstrap. During one of the lessons early on, we introduced using programming to make shapes of all different sizes and colors. He was interested in the idea of making shapes, which aligns with his enjoyment of design. However, when he reached a mathematical or programming concept he needed to figure it to move forward, he would stop. So as soon as he reached the part he did not like, he gave up. During the last lesson, the students spent most of the time working on creating an image of a flag that they had designed. After the workshop, his teacher told us that he was “super pumped to do the flag project”. His level of interest in that lesson was definitely higher than in all the previous ones. He drew a picture of the flag he wanted to design right away, but then was unable to get started on the code. The TA spent a lot of time with him in the beginning to help him get started. It required a lot of prodding, as usual, but when he saw the flag he had drawn start to take form on the computer, he became excited about the programming aspect of it. For the rest of the lesson, he was actively asking questions. He really wanted to accomplish getting the final version of the flag he had created on the computer, see Figure 1.

Nicole

Nicole was an active and engaged participant in all classes. She was always the first to raise her hand to answer questions and she would usually answer the questions correctly. She would work hard during the class and ask thoughtful questions of the facilitator and the TA. During the final flag project (see Figure 1), when the possibility of saving her code to work on later, she seemed excited about continuing her work. From our interactions with her, she appeared to be really enjoying working with Bootstrap and learning to program. During our discussion with Nicole’s teacher, we learned that she will perform in school just because she wants to please her teachers. Nicole thought it was a waste of time to learn to program because she does not feel computer science is important to her future career.

Anita

Our initial observations of Anita from the first day indicated that she was really excited about learning programming. She seemed fascinated by using statements on the computer to build cool images with shapes, exclaiming “Ooh cool!” the first time she was successful. Her teacher informed us later on that she has a “deep down enjoyment” for mathematics. During the second lesson, Anita was sharing a computer with another student and they were supposed to be pair programming. The student she was sharing with struggled with mathematics and also had a hard time focusing. We observed a definite decrease in her excitement level that day. She did the minimum of what was instructed, but was not as interested or explorative as she was on Day 1. Over the next couple of lessons, the decreased level of excitement remained constant, although she did always seem to be keeping up and understanding. The second-to-last lesson a new student, Xiomara, was added to her table. Xiomara had missed all the previous lessons, and did not know what was going on. It was clear from the beginning that Xiomara had no intention of trying to get caught up or learn what was even going on. From this point forward we noticed that Anita was portraying a disinterest in programming which coincides with how her teacher describes her as “putting up a front for her peers” and is often influenced by those around her.

3.3 Data Collection

On the first and last day we administered a pre and post mathematics test and survey. The mathematics pretest was designed by the teacher based on the mathematics curricula that she felt most appropriate. She based the questions on the common core standards that were covered by the Bootstrap units we would be covering. The mathematics post test was was similar to the pretest in terms of types of questions, but with slight modifications. The surveys measured student's interest and belief in ability in mathematics and computer science. The post-survey was exactly the same as pre-survey, but with four additional questions.

Mathematics Test

The mathematics tests included fraction addition, creating and solving linear equations, determining slope and y-intercept and algebraic word problems (Appendix A). On average, both the number of questions attempted and the score increased between the pre-test and the post-test. The increase in problems attempted was 13% and the increase in score was 67%.

While the mathematics tests show improvement, we do not feel that the results are conclusive. The mathematics test was designed by the teacher to be aligned with the standards she had chosen based on the Bootstrap standards list. However when comparing the actual lesson plans with the test, there was not much overlap. Our lessons did not cover fractions, slope and y-intercept and we did not effectively make the connection between algebraic equations and Bootstrap functions.

	Pretest		**Post test**			
	Attempts	Score	Attempts	Score	Attempts	Score
Student	out of 12	out of 12	out of 10	out of 10	increase	increase
Todd	9	2	9	5	17%	200%
David	6	2	10	2	50%	20%
Nicole	12	6	10	6	0%	20%
Anita	5	3	5	2	17%	-20%
A	6	2	9	6	44%	260%
B	12	10	6	4	-67%	-52%
C	11	5	9	3	-2%	-28%
D	10	4	7	4	-19%	20%
E	5	2	8	4	48%	140%
F	6	4	9	7	44%	110%
Avg Raw	8.2	4	8.2	4.3		
Avg %	68%	33%	82%	43%	13%	67%

Table 0: Pre and post test scores from each student. Students who we discuss specifically are named.

Survey

We administered a pre and post survey based to measure student's interest and belief in ability in mathematics and computer science. We had issues getting meaningful responses from the surveys because the students were not answering the reversed questions in correspondence with the non-reversed questions (i.e. Todd strongly disagreed with both statements: 'I find math enjoyable' and 'Math just doesn't appeal to me'). Of the ten students in the class, four of them had usable full survey results. The results are displayed in Table 1. The results are not especially informative. We do not see a dramatic change in any of students and if anything the students are slightly less interested in learning mathematics and computer science than before. The telling result is that only one of the students was interested in continuing to use Bootstrap and two of the four students were interested in learning more computer science. However all the students are excited to learn more mathematics.

Student	A		B		C		Anita	
	Pre	Post	Pre	Post	Pre	Post	Pre	Post
I find math enjoyable.	5	5	5	5	4	4	5	5
Math just doesn't appeal to me. (Reversed)	1	1	1	1	3	3	1	1
I enjoy computer computer science.	5	5	4	4	3	2	3	3
I like working on math problems.	5	5	4	4	1	2	4	4
I don't like computer science. (Reversed)	1	1	1	2	3	4	3	4
I think I did well on the quiz.	4	5	5	5	2	2	2	1
I felt confident in my answers.	4	5	5	5	2	2	2	1
I felt that I was doing poorly on these problems. (Reversed)	1	1	2	1	3	3	5	4
I do well in math.	5	5	4	4	4	3	4	4
Math is useful in everyday life.	5	5	5	5	5	1	5	4
Computer science is useful in everyday life.	5	4.5	4	3	4	3	4	2
I don't think math is useful for me.	1	1	1	1	1	5	1	3
I don't think computer science is useful for me.	1	1	2	2	3	4	3	3
Programming is interesting.		5		4		4		3
It was a waste of time to learn to program. (Reversed)		1		2		3		3
Learning to program was enjoyable.		5		5		4		3
Do you want to continue using Bootstrap to learn math?		Yes		No		No		No
Do you want to learn more computer science in the future?		Yes		Yes		No		No
Are you excited to learn more math in the future		Yes		Yes		Yes		Yes

Table 1: Select survey results. On the Likert Scale we are using 1 is strongly disagree with the statement and 5 is strongly agree with the statement.

Section 4 Discussion of Results

Due to the inconclusiveness for several students on both the pre- and post-test and the pre- and post-survey, the discussion section will predominantly be based on ethnographic observations of the four students from the class described in the data collection section.

Todd

Todd's development over the course could support the premise of Eccles et al that positive expectancies or feeling of competence can increase the perceived value in activities. This is supported both by Todd's immense happiness after making the "coolest" image and by considering his relatively sustained interest and participation throughout the rest of the course. As such, the ethnographic observation of Todd could let us suggest that for some struggling mathematics students, the creative power of Bootstrap can help them positively increase their attitudes towards functional programming for mathematics. Unfortunately Todd's survey results were inconclusive which does not let us conclusively justify our research hypothesis that functional programming positively increases competence and attitudes towards mathematics.

David

In general David had a lack of interest in school which was demonstrated throughout the course. However David was interested in design which increased his interest and concentration when making shapes. This observation could support Hulleman et al's premise that generating personal conceptual outcomes of the course material even in subjects one dislike can positively increase attitudes towards, what in this case was, functional programming for mathematics. This point was emphasized at the end of the Flag project when David was very excited to build something of value to him. However as David unfortunately had inconclusive survey results and when considering his observed lack of persistence in finding solutions when he gets stuck, it is not possible to conclusively justify our research hypothesis that functional programming positively increases competence and attitudes towards mathematics. Although it would be interesting to see if, with more time, David's behavior would continue.

Nicole

During our discussion with Nicole's teacher, we learned that she will perform in school just because the authority figure (i.e. teacher) has told her to do so. When looking at her survey results (see Table 2), we saw a different story from the interested girl we had thought she had been. Her interest in computer science actually decreased. When conducting experiments, we, as researchers, need to be cognizant that observations may not align with true feelings. Hence it is not possible to conclusively justify our hypothesis that functional programming positively increases competence and attitudes towards mathematics.

Statement	Pre	Post
I enjoy computer science	Disagree	Strongly Disagree
I don't like computer science	Agree	Strongly Agree
I think computer science is useful in everyday life	Disagree	Strongly Disagree
I don't think computer science is useful to me	Agree	Strongly Agree

Table 2: Subset of survey results from Nicole

Anita

Anita showed enthusiasm from the first day of the project. This could suggest that Anita was excited due to her ability to use what she had learned to build a shape she personally found 'cool'. However when discovering through conversations with her teacher that Anita had a great enjoyment for mathematics, her enthusiasm cannot justify that functional programming increases task engagement and positive performance expectations towards mathematics. However an important factor to consider when working with middle school students is that the outward behaviour of this age group is heavily influenced by their peers. Her teacher said that she "puts up a front for her peers" and is often influenced by those around her. Upon reflection of this and our observations from her, we wonder if it is really true that she doesn't like computer science. It could be possible that her survey answers were influenced by Xiomara because they were talking while filling out the surveys. Something that definitely needs to be considered when observing students in this age category is that their outward behavior can be heavily influenced by their peers. In summary, the ethnographic observations of Anita's experience with Bootstrap does not conclusively justify our research hypothesis that functional programming positively increases competence and attitudes towards mathematics.

Statement	Pre	Post
I enjoy computer science	Neutral	Neutral
I don't like computer science	Neutral	Agree
I think computer science is useful in everyday life	Agree	Neutral
I don't think computer science is useful to me	Neutral	Agree

Table 3: Subset of survey results from Anita

4.1 Evaluation of Methodology and Lessons Learned

Neither the ethnographic observations or the surveys and test results suggested that Bootstrap or functional programming increases competence and positive attitudes towards mathematics. However our findings, due to the short time period in which we evaluated our participants and the fact that we were working with a subset of the Bootstrap curriculum, are not sufficient to challenge the established literature on Bootstrap's ability to facilitate students conceptualization of mathematics. Although our findings suggest that for some participants Bootstrap increased positive attitudes towards functional programming, the relatively short timespan of the project made it difficult to determine whether this positive change in attitude towards functional programming would have transferred over to mathematics. Measuring the effect Bootstrap had on competence levels in mathematics also was difficult due to the limited timespan of the project.

There are a few key limitations beyond the limited timespan of the project that could have contributed to these inconclusive results:

1. The pre and post surveys were largely inconclusive. This could potentially be caused by the wording of the survey as it might have been too complex or vague. Alternatively it could be caused by the time the participants conducted the survey which was at the end of class which might have caused some students to rush through it. A third possible cause is that the students lacked motivation to complete the surveys thoroughly. The participants understood that they were not being graded and that the project had no effect on their normal class evaluations.

2. Even though the post test results went up, we don't believe they prove anything.This can be attributed to a few methodological errors the research team made. First, the post test was conducted at the end of the class where the students were tired and unwilling to give their best effort. The lack of effort could also be attributed to the students knowing they were not being graded on the test and that the results had no effect on their normal class evaluations. Secondly, the test should have been based more directly on the bootstrap curriculum and specifically the units that were being taught. Although both the tests and Bootstrap were based on the common core curriculum, the test were not aligned sufficiently with the topics covered during the project which made it hard to evaluate what effect the Bootstrap curriculum have on mathematical competency levels.

3. Another key limitation of the project was the inability of the research team to effectively control and organize the group dynamics within the classroom. Creating partners that do not distract each other with non-course related activities is essential for the individual student to progress through the Bootstrap curriculum. Additionally, none of the research team members had professional experience with teaching a middle school age group which made it difficult at times to find ways of motivating the students to continue working and progress through the curriculum. Additionally the students in the class were not used to a lecture format of learning

which is why in future projects, acquiring a professional teacher, trained in Bootstrap, to handle the instruction part is essential to capture the attention of the students.

4. A last lesson learned is that with a Middle School age group, it is difficult to predict how the lesson plan will go. More tasks and activities should have been made available to keep students who progress at different levels motivated. This could potentially have incentivized students to work hard on each task so they could progress to the more advanced tasks. However it would be important to have these tasks open ended enough so the students can create something they themselves find interesting or of value to them. Again this reflects the lack of teacher training of the research group. Points three and 4 are evidence of the importance of the claims of Sherin and van Es of the importance of noticing and adapting instruction during a lesson. Our instructors did not have this expertise, and often our teaching was ineffective but we had no ability to understand in the moment what was happening and how to adjust.

Section 5: Conclusion/Future Work

It is evident from our findings that our observations could not conclusively justify our hypothesis that Bootstrap positively increases competence and attitudes towards functional programming for mathematics. However the the ethnographic observations do produce a basis for suggesting that Bootstrap could, for some students, positively increase attitudes towards functional programming but due to the limited timespan of the project and inconclusive survey and test results it is not possible to conclude that this positive change in attitude would change students attitude and competence towards mathematics. For future projects, the lessons learned would definitely have to be incorporated to produce more conclusive results.

References

1. Eccles, J. S., Harold, R. D., “Gender Differences in Sport Involvement: Applying the Eccles’ Expectancy-Value Model, *Journal of Applied Sport Psychology*, Vol 3, 1991

2. Eccles, J. S., Wigfield, A., “In the Mind of the Actor: The Structure of Adolescents’ Academic Achievement Related-Beliefs and Self-Perceptions”, *Personality and Social Psychology Bulletin*, 1995

3. Emerson, R. M., “Writing Ethnographic Fieldnotes”, University of Chicago Press, Chicago, 1995

4. Felleisen, Matthias, et al. "A functional I/O system or, fun for freshman kids." *ACM Sigplan Notices*. Vol. 44. No. 9. ACM, 2009.

5. Felleisen, Matthias, and Shriram Krishnamurthi. "Viewpoint Why computer science doesn't matter." *Communications of the ACM* 52.7 (2009): 37-40.

6. Hulleman, C. S., Godes, O., Hendricks, B. L., & Harackiewicz, J. M., “Enhancing Interest and Performance With a Utility Value Intervention”, *Journal of Educational Psychology*, 11 Oct 2010

7. National Council of Teachers of Mathematics. (2000). Principles and standards for school mathematics. Reston, VA.

8. Schanzer, Emmanuel, et al. "Transferring skills at solving word problems from computing to algebra through bootstrap." *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. ACM, 2015.

9. Sherin, M. & van Es, E. (2005). Using Video to Support Teachers’ Ability to Notice Classroom Interactions. Journal of Technology and Teacher Education. 13 (3), pp. 475-491. Norfolk, VA: Society for Information Technology & Teacher Education.

10. Van Es, E.A, & Sherin, M.G. (2002). “Learning to Notice: Scaffolding New Teachers’ Interpretations of Classroom Interactions”. Jl. of Technology and Teacher Education 10(4), 571-596

11. Wright, Geoff, Peter Rich, and Robert Lee. "The influence of teaching programming on learning mathematics." *Society for Information Technology & Teacher Education International Conference*. Vol. 2013. No. 1. 2013.

12. Yoo, Danny, et al. "WeScheme: the browser is your programming environment." *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*. ACM, 2011.

Appendix A: Pre/Post test

Bootstrap Pretest Name:

Solve the following problems using any strategy you would like! Make sure you show us your thinking/work.

1. Moe's Bagels sells 12 bagels for $6. How many bagels can they buy for $20?

2. Write an ***equation*** which relates the cost of the bagels, *C*, to the number of bagels, *n*.

C = _____

3. What is the **slope** (rate) and the **y-intercept** (starting cost) of your equation?

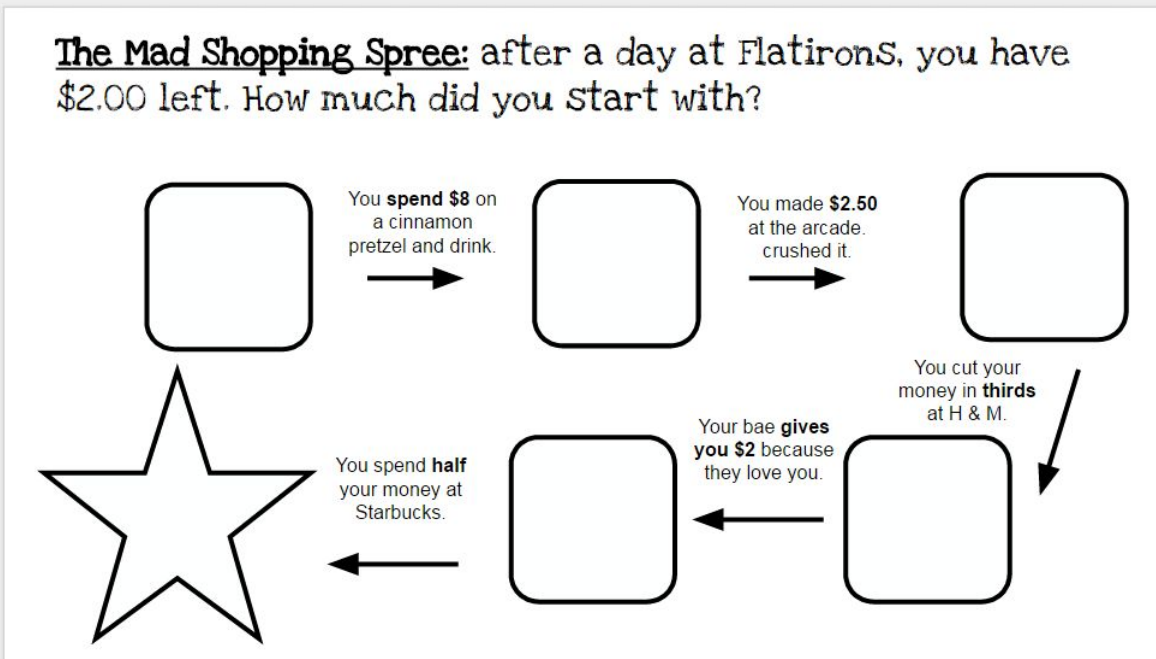
Slope = _____ **Y-Intercept** = _____

---

4. What is ½ + ⅛ ? Choose the correct answer below.

a) 2/8 b) ⅜ c) ⅛ d) 1/12

5.



6. Choose equation to solve for *x*.

a) 11 = 3x + 2 *b) d/4 + 3 = 5* *c) 13x - 15x + 8 = -4x + 2 - 24*

Bootstrap Posttest Name:

Solve the following problems using any strategy you would like! Make sure you show us your thinking/work.

1. Moe's Bagels sells 18 bagels for $9. How many bagels can they buy for $30?

2. Write an ***equation*** which relates the cost of the bagels, *C*, to the number of bagels, *n*.

C = _____

3. What is the **slope** (rate) and the **y-intercept** (starting cost) of your equation?

Slope = _____ **Y-Intercept** = _____

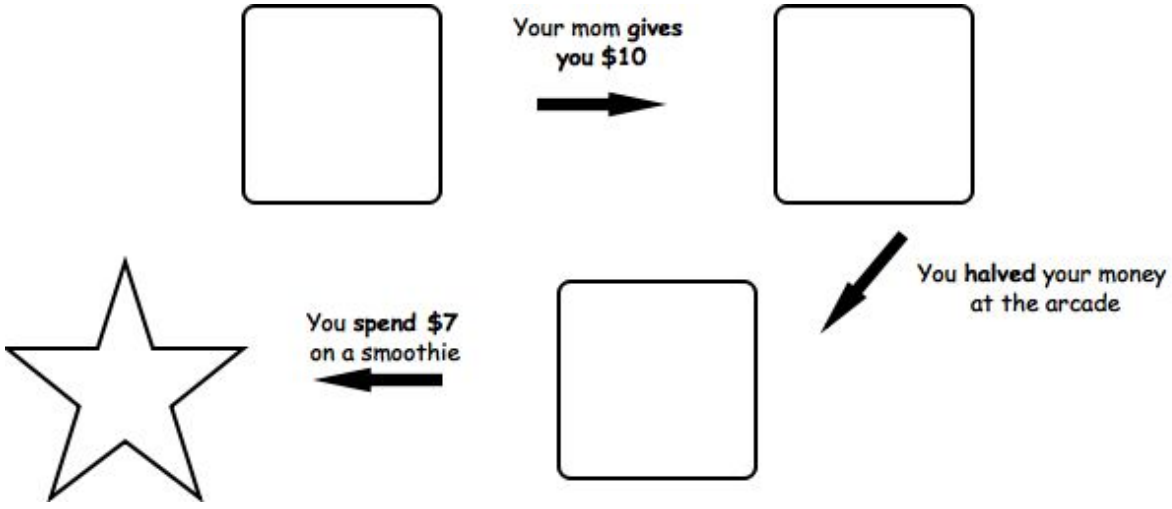
---

4. What is 1/3 + 1/4 ? Choose the correct answer below.

a) 7/9 b) 2/7 c) 7/12 d) 1/12

---

5. A day at the Mall: After a day at the Flatirons, you have $3 left. How much did you start with?



6. Solve for x in the following equation.

12 = 5x+7

Appendix B: Survey Questions

Pre/Post Survey Questions modified from (Hulleman et al)
Participants responded to all self-report items in this study on a 5-point Likert-type scale from 1 (*strongly disagree*) to 5 (*strongly agree*).

Pre Test

Initial Interest

I find math enjoyable.
Math just doesn't appeal to me. (Reversed)
I enjoy computer science.
I like working on math problems.
I don't like computer science. (Reversed)

Performance Expectations

I think I did well on the quiz.
I felt confident in my answers.
I felt that I was doing poorly on these problems. (Reversed)
I do well in math.

Utility Value

Math is useful in everyday life.
Computer science is useful in everyday life.
I don't think math is useful for me.
I don't think computer science is useful for me.

Post Test Additions

Situational Interest

Programming is interesting.
It was a waste of time to learn to program. (Reversed)
Learning to program was enjoyable.

Maintained Situational Interest

Do you want to continue using Bootstrap to learn math? (Yes/No)
Do you want to learn more computer science in the future (Yes/No)
Are you excited to learn more math in the future (Yes/No)

Appendix C: Lesson Plans

Lesson 1: Bootstrap Hour of Code

Facilitator/Teacher: Alex, Teaching Assistant: Ole, Observer: Anna

- Intros (Anna):
    - Who we are
    - Why are we here
    - What we are going to be going
    - Emphasize participatory research:we're not here just because we want to be math tutors, we're here because we want to learn what works and what doesn't with you. "We want your feedback with these lessons - what was cool, what was boring? Did you you have any 'aha' moments where you learned something that you were totally stuck on before?"
- Pre-Test
- Pre-Survey
- Hour of Code
    - Expression: a computation written in the rules of some language (such as arithmetic, code, or a Circle of Evaluation)
        - 2, 3 or 2+3
    - WeScheme Editor
        - Definitions Area on the Left
            - Write Expressions and click to run button to see result
        - Interactions Area on the Right
            - Write Expressions and hit enter to see results
        - Reading and Writing Code
            - Function (+, star)
            - Inputs (what the function needs to produce a result (output)
                - Numbers - evaluate to themselves
                - Strings (words with quotes around them, how the program reads words) - evaluate to themselves
            - Output (result from applying function to inputs: seen in the interaction window)
            - Need Parenthesis for evaluation
            - Circle of Evaluation
                - **Rule 1**: Each circle must have one function, which goes at the top of the circle.
                - **Rule 2**: The inputs are written below, in order from left to right.
                - Examples (Circles and Code)
                    - (+ 2 3)
                    - (star 50 “solid” “red”)
            - Picture Example

- (overlay (scale 0.5 (bitmap/url https://static01.nyt.com/images/2016/02/29/sports/basketball/STEPHCURRY/STEPHCURRY-articleLarge.jpg")) (rectangle 350 220 "solid" "gold"))

Lesson 2: Unit 1 Bootstrap Curriculum

http://www.bootstrapworld.org/materials/spring2016/courses/bs1/units/unit1/index.html

Unit 1 of the Bootstrap curriculum officially takes 85 minutes however we have created a revised version that will be 50 minutes.

Unit 1 mainly consists of three different stages. Together these stages introduce computational thinking on how video games are built along with how the different elements on the screen interact.

Stage 1 - *"Dissecting a Demo with X,Y Coordinates" 10min*

Each student will be asked in the preceding class to bring a picture of a scene in a video game they like or have played. To start we will all be using the NinjaCat video game (http://www.wescheme.org/run?publicId=sggzRzgU5T). The students will be asked to identify what characters exist in the screenshot of the NinjaCat game. All the characters will be listed in a table and we will add to the table what changes with every character when the game happens. Specifically we will focus on how each character changes considering X,Y coordinates. For example: Can NinjaCat move up and down in the game? Can she move left and right? So what's changing: her x-coordinate, her y-coordinate, or both? What about the clouds? Do they move up and down? Left and right? Both?

Stage 2 - *"Order of Operations for Numbers" 20min*

Math is a language, just like English, Spanish, or any other language. We use nouns, like "bread", "tomato", "mustard" and "cheese" to describe physical objects. Math has *values*, like the numbers 1, 2, or 3, to describe quantities.

Humans also use verbs like "throw", "run", "build" and "jump" to describe operations on these nouns. Mathematics has *functions* like addition and subtraction, which are operations performed on numbers. Just as you can "slice a piece of bread", a person can also "add four and five".

Mathematicians didn't always agree on the order of operations, but now we have a common set of rules for how to evaluate expressions. The pyramid on the right summarizes the order. When evaluating an expression, we begin by applying the operations written at the top of the pyramid (multiplication and division). Only after we have completed all of those operations can we move down to the lower level. If both operations are present (as in 4+2−1), we read the expression from left to right, applying the operations in the order in which they appear.

Let's code!

We will be using WeScheme for our coding section. WeScheme is a free website that can be accessed on any computer with internet access. The WeScheme editor has a 'definition' and an 'interaction' section. You define your function in the 'definition' section and call it in the 'interactions' section with your desired values to perform the needed computation. http://

www.wescheme.org/openEditor

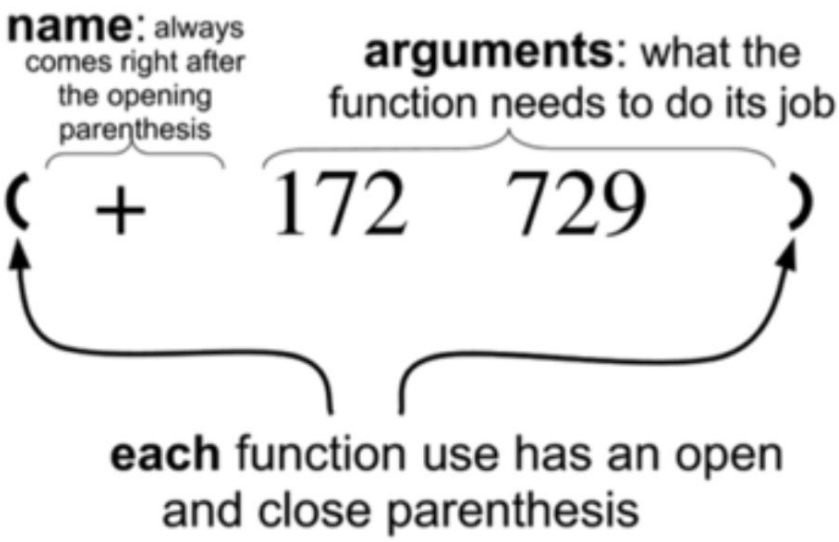
The Circles of Evaluation are also easy to convert into computer programs. To translate a Circle of Evaluation into a program, begin with an open parenthesis `(`, and then the function written at the top of the circle. Then translate the inputs from left to right in the same way, adding a closing parenthesis `)` when you're done. This process gives us the second rule for ***expressions***:

Code Rule 2: Each open parenthesis is followed by one function, then by one or more legal expressions, and finally by a closing parenthesis.

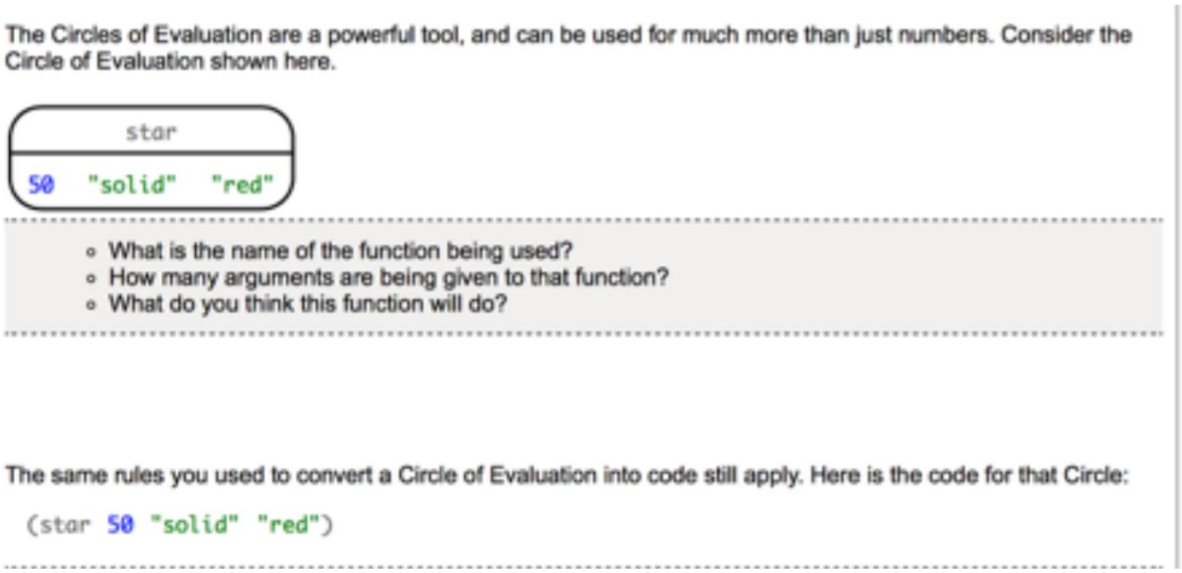
Here is the code for this Circle of Evaluation: `(- 4 5)`

See what happens when this code is entered into the Interactions area. Press the Return key to evaluate the program. You should see -1 as an answer.

All of the expressions that follow the function name are called *arguments* to the function. The following diagram summarizes the shape of an expression that uses a function.



Stage 3 - *"Order of Operations for Shapes" 20min*



Copy the above code into your definitions section. What happens when you click 'Run'? What happens if you change the number to 100? What happens if you change the second argument in your function to "outline"?

See what happens if you change 'star' with other shapes like 'circle' or 'square'?

Lesson 3: Unit 2 Bootstrap Curriculum

Warm-up (Do together on the board)

1. `3` is a…
    a. Number
    b. Function
    c. Variable
2. `+` is a …
    a. Number
    b. Function
    c. Variable
3. Draw the circles of evaluation for (3*4)+5
4. What is the corresponding Racket code? (Should probably decide on Racket vs. Scheme to stay consistent) `(+ (* 3 4) 5)`

Strings and Images

- More than just `Numbers`
    - Two new types: `Strings` and `Images`
    - `Strings` are surrounded by quotations (i.e. "solid")

    - `Images` are shapes and/or pictures
- Try it out!
    - `(star 50 "solid" "purple")`
    - `(star (+ 1 3) "outline" "blue")`

Contracts

- From our experiments, we have seen that difference functions take different inputs
    - `star` takes in a `Number` and two `Strings`
    - `+` takes in two `Numbers`
- The expected inputs of the function are its domain
- The expected outputs of the function are its range
- Contracts for Functions (let's the programmer know how to use them)
    - Name
    - Domain (number of things and their types)
    - Range (number of outputs and their types)
    - If we think of a language as a collection of lego pieces, the Contracts are like the tabs and slots that tell us how each piece can connect.
    - `; star: Number String String -> Image`
    - `; +: Number Number -> Number`
    - Try it out!
        - `; rectangle: Number Number String String -> Image`
            - What is the name, domain, and range?
        - Write the contract of `*`
- Function composition puts together multiple functions by matching up the domain and ranges
    - `(+ (* 3 4) 5)`
    - Like a puzzle where we match up the domain of one function with the range of another
    - Try it out!
        - Draw a circle of radius 25 using only the number 1-10
            - `(circle (+ 10 (+ 10 5)) "solid" "green")`
            - `(circle (* 5 5) "solid" "green")`
- Error Messages
    - Rectangle example
        - `(rectangle 50 "solid" "red")`
    - Circle example where fill and color are switched
        - `(circle 10 "green" "solid")`

Lesson 4: Unit 3 Bootstrap Curriculum

Review (5 min) worksheet:

Circles Completion

Math Circle of Evaluation

(3 * 7) - (1 - 2)

3 - (1 + 2)

3 - (1 + (5 * 6))

(1 + (5 * 6)) - 3

Warm-Up (5 min) on board:

;mystery : Number String String String -> Image

1.) Underline the Name of this function

2.) Circle the Domain of this function

3.) Put a box around the Range of the function

4.) Image is a ...

a. Function b. Type c. Value

5.) 6281 is a ...

a. Function b. Type c. Value

Definitions : (15 min)

Can give names to values using define statements:

(define *pinkstar* (star 50 "solid" “pink"))

Solid pink star of size 50 is stored in value *pinkstar*

Definitions are useful because you can use them in other expressions. For example,

(define pinkstar (star 50 "solid" “pink”))

(define purpstar (star 100 "solid" “purple”))

pinkstar and purpstar can be used in overlay function:

(overlay pinkstar purpstar)

Try it out!

Defining Functions (20 min) :

Define a function gt, which takes a Number and produces a solid, green triangle of the given size.

Whats values will always be the same with each call of the function? (solid, green, triangle)
What value could be different? (size)
What is Name of function? (gt)
What is the Domain of function? (Number) What is the Range? (Image)

;gt Number -> Image

Go through Examples, and how they help see what is variable and what is constant

1.Copy everything that stays the same (everything that wasn’t circled) in one of your EXAMPLE lines (onto paper or into your editor)
2.In place of each circle, write the label you gave to that circle
3.Change EXAMPLE to define

1.(EXAMPLE (gt 50) (triangle 50 “solid” “green”))
2.(EXAMPLE (gt 100) (triangle 100 “solid” “green”))
3.(define (gt size) (triangle size “solid” “green”))

These steps are known as DESIGN RECIPE

Go through example of making a picture of a dartboard, by defining functions for the two different colored circles on WeScheme.

Introduce put-image function:

;put-image Image Number Number Image -> Image

put-image is just like overlay, except it also takes in x,y coordinates as inputs.
The Image isn’t necessarily centered on top of another Image
Can make a lot more interesting shapes!

Go through example of making Colorado flag using put-image on WeScheme

Lesson 5: Unit 6 Bootstrap Curriculum

Follow Unit 6 lesson plan (http://www.bootstrapworld.org/materials/spring2016/courses/bs1/units/unit6/index.html), until it gets to the game, replace with a bounce program.

Video for teachers: https://www.youtube.com/watch?v=OMQO66wWqjk

Review - 15 minutes (this seems long!)
What datatypes have you seen so far? Can you think of Number values? String values? Image values? What functions have we seen so far?

Expressions are functions with values. Example plus. Give me a function that evaluates to a number - give me an example expression (not the answer or output - just the expression) . Give me a function that evaluates to an Image - give me an expression.

Unit 6 Warmup activity sheet.

Booleans - 10 minutes

- Teach <, >, =
- Booleans on true or false
- string=?
- Contracts

Expressions can have sub expressions (> (+ 2 4) (- 8 3)) - what the output?
Onscreen 1.0: Sam the butterfly - 25 minutes (too long?)

- Pages 19-21 workbook
- Cage.rkt http://www.wescheme.org/view?publicId=1pLdvPhmYa
  (kari's changes)
- Make a safe left, safe top safe bottom

and/or - 10 minutes?

- Page 22 workbook (or true false true false)

Onscreen 2.0: 20 minutes (skip game)

- Use functions safe-left, safe-right - Why?
  - Good code design - simple functions -> complex functions (testing)
  - Names better than formulas - easier to read