Unit 10
Bridging to Algebra

Unit Overview

Students translate from Racket into Algebra, and back. They then apply the Design Recipe to solve common word problems from Algebra texts.

Agenda

5 min	Introduction
10 min	Defining Values
20 min	Defining Functions
30 min	Design Recipe
30 min	Harder Problems

Learning Objectives:

- Take what they've learned in the programming domain, and apply it to the algebraic domain.

Product Outcomes:

- Students rewrite their games in Algebraic notation and solve algebra problems.

State Standards See our Standards Document provided as part of the Bootstrap curriculum.

Length: 90 minutes

Materials and Equipment:

- Computers w/ DrRacket or WeScheme
- Student workbook folders - in pairs! - with names on covers.
- Pens/pencils for the students, fresh whiteboard markers for teachers
- Class posters (List of rules, basic skills, course calendar)
- Language Table (see below)
- Open the Algebra Translation [] file on students' computers, but turn the monitors off.

Preparation:

- Write agenda on board
- "Algebra Translation" [DrRacket | WeScheme] preloaded on students' machines, with monitors off.

Types	Functions
Number	`+ - * / sq sqrt expt`
String	`string-append string-length`
Image	`rectangle circle triangle ellipse radial-star scale rotate put-image`
Boolean	`= > < string=? and or`

Introduction (Time 5 minutes)

- So here we are, at the final unit of Bootstrap.
- You've learned a lot about computer programming, and you've used many of the concepts you've picked up along the way to build your videogames.
- It turns out that these concepts show up in a lot of places outside of this class, too. If you take another programming class, for example, you will still find yourself using functions and variables. You will still find yourself wrestling with Numbers, Strings, Images, and Booleans. If there is one thing I hope you take with you, it is the Design Recipe.
- Today we're going to explore how the Design Recipe can be used for a totally different class – Algebra.
- It turns out that math is actually a programming language! Arithmetic (adding, subtracting, multiplying and dividing) is just like in circles of evaluation, where you could use functions but didn't know how to define them.
- Algebra, on the other hand, is where you start defining functions and start using them to solve problems. You've been doing this with code throughout the entire class!

Defining Values (Time 10 minutes)

- *For this activity write all Racket expressions on one side of the board, and all algebra expressions on the other.*
- In our programming language, we defined values this way: `(define x 4)`
- I can even define values in terms of values that came before: `(define y (+ x 10))`
- I want a volunteer who can define another value for me.
- *Copy their answers as you go. Encourage students to define values in terms of other ones.*
- In math class, you define values using the equals sign: `x = 4`
- *Write this on the other side of the board.*
- And, as with our programming language, you can define values in terms of other values: `y = x + 10`
- Turn to page 35 in your workbooks. You will see a bunch of value definitions written in code - take 2 minutes to convert this into math. GO!

Defining Functions (Time 20 minutes)

- *Write a simple function on the board:*
- `(define (f x) (+ x 1))`
- You also know how to define functions like this one. Raise your hand if you can tell me the name of this function?
- How many variables does it take? What is the name of that variable?
- What does the function do to the variable `x`?
- Can you define a function `g`, which takes in a variable `q` and multiplies it by 20.
- *Have students define 2-3 other, simple functions.*
- Can you explain what this function does?
- To translate these functions into algebra, we do something similar to what we did with the values. We remove the `define`, and replace it with an equals sign: `f(x) = x + 1`
- Who can translate the function `g`?
- `g(q) = q * 6`
- *Have students translate the rest of the functions, listed on Page 37.*
- Okay, so you're pretty good at translating from code to math. Do you think you can translate from math back into code?
- *1...2...3 Monitors ON!!!*
- On your computers, there are five examples of functions written in algebra. Under each one, I want you to convert the algebraic function into code. You will have five minutes - GO!
- *Countdown, review as needed.*

Design Recipe (Time 30 minutes)

- Of course, the real power of programming isn't how well you know the language. It's about how well you can use it to solve problems! In this class, you've learned about a powerful tool that helps you take word problems on paper and turn them into functions on the computer: the Design Recipe!
- It turns out that the Design Recipe can also be used to help you solve word problems in algebra, too!
- Just like we have in the past, we're going to test out your Design Recipe skills...but this time, you'll be using it with Algebra, instead of Code.
- For Round 1, let's take a problem you've seen before, and walk through it the way we would in an algebra class. Turn to Page 36 in your workbooks.
- "A rocket is flying from Earth to Mars at 80 miles per second. Write a function that describes the distance `D` the rocket

has traveled, as a function of time `t`."

- Where do we start? Give students some time to think, or make suggestions.
- Well, we want to write a function - so what's the first step in the Design Recipe? We need to make a contract!
- Every contract has three parts. What are they?
- As usual, the first two parts of the contract sometimes have hints to them in the problem itself. Take a minute, and see if you and your partner can underline the hints for the Name and Domain of this function.
- So what did you find? The problem wants us to find out the distance the rocket has traveled. So we could call the function `Distance`, or `D` for short. Sometimes we can choose our own name, and sometimes the problem tells us which name it wants. Does this one tell us whether it should be `D` or `Distance`?
- What about the Domain? What kind of thing goes into the distance function? Is it how many people are on the rocket? How many stops it makes? Did anybody find the hint in the word problem?
- It's the number of seconds the rocket has been traveling! What about the Range? What kind of thing is distance? It's the number of miles the rocket has traveled! In algebra, we can actually be a little more descriptive than in programming, so we can simply write `; D : seconds -> miles`
- Let's also make a little note to ourselves, to remind us of what this function is all about.
- `; D : seconds -> miles`
- `; A rocket travels 80 miles every second: how far it gone?`
- Raise your hand: Can you tell me the next step in the Design Recipe?
- We need to give examples for the function. On the computer, we would start out by writing `(EXAMPLE...)`, followed by an example of using `d` with some number of hours and then the code that gives us a distance. In Algebra, we don't need to write the word `EXAMPLE`, but everything else we know still applies.
- Suppose the rocket has been traveling for just one second. That means our distance function `D` has what for an input? One! What would I write if it had been traveling for three seconds? Ten? Write these on the board as students answer...
- `; D : seconds -> miles`
- `; A rocket travels 80 miles every second: how far it gone?`
- `D(1)  = ...`
- `D(3)  = ...`
- `D(10)  = ...`
- When the rocket has been moving for a second, how many miles has it gone?
- *Have students discuss, and explain their answers. Fill in the table as you go.*
- `; D : seconds -> miles`
- `; A rocket travels 80 miles every second: how far it gone?`
- `D(1)  = 80*1`
- `D(3)  = 80*3`
- `D(10)  = 80*10`
- Can you come up with more examples?
- Okay, so now we've done some examples. Raise your hand if you can tell me what we do next?
- Now we circle everything that's changed! In this case, that's just the second number - the one thay gets multiplied by 80. What we need now is a good variable name. What does that number mean? Think back to what was underlined on your paper, when we talked about what goes into the function. What should we call it?
- *By now, students should be able to figure out the variable name themselves. However, it's worth pointing out that some word problems (like this one!) will tell students what to call the variable.*
- Okay, so we've got our contract, our examples, and our variable name. We're ready to define the function! Just as we always have, we copy our examples - replacing the circles items with the variable names.
- `D(t) = 80*t`
- For Round 2, I'm going to give you a slightly different problem, and see if you can figure out how to write the function. Turn to Page 37 in your workbooks.
- "A rocket is traveling from Earth to Mars at 80 miles per second. Write a function that describes the time the rocket has been traveling, as a function of distance."
- Take a minute with your partner, and see if you can underline all the hints in this word problem.
- *If necessary, point out that this is the same relationship between distance and time as before, only now we want be able to see the relationship from the opposite direction: time in terms of distance, rather than distance in terms of time.*
- What's the first step? Write the contract!
- `; time : miles -> seconds`
- What comes next? We write the function, right? NO! We need to write some examples!
- *Have students volunteer some examples, making sure they are written in algebra (students may choose their own function name)*
- `; time : miles -> seconds`
- `; A rocket travels 80 miles every second: how long has it been flying?`
- `time(0) = 0/80`
- `time(100) = 100/80`

- `time(80) = 80/80`
- `...`
- Now see if you can finish this off with your partner, choosing your variable name and then writing out the function. If you can complete this step in two minutes, your team will earn a point. Ready, set...GO!
- *Countdown, assign points, and review.*
- Once your function is set up, it's easy to just plug in values and get answers back. With most word problems, the hard part is setting up the function in the first place.
- Luckily, the Design Recipe makes setting up that function a lot easier! We've just used it to set up two different functions, which could be used to give us answers in terms of distance or time.
- Suppose we had a word problem that wanted to know how far the rocket traveled in 6 seconds: which one would we use? What if we wanted to know how long it takes for the rocket to go a thousand miles? What if I knew the train left at 1pm, and I wanted to know what time it arrives in Chicago, 800 miles away? Would I want my time function, or my distance function? Have students explain their answers to each.
- Let's make it more interesting...

Harder Problems (Time 30 minutes)

- Turn to Page 38 in your workbooks.
- "A rocket leaves Earth, headed for Mars at 80 miles per second. At the exact same time, an asteroid leaves Mars traveling towards Earth, moving at 70 miles per second. If the distance from the Earth to Mars is 50,000,000 miles, how long will it take for them to collide?"
- What is this problem asking us for? The time it takes for the rocket to collide with the asteroid
- What we want is a function that will tell us how many seconds until a collision occurs, given how far apart the rocket and asteroid are.
- What is a good name for this function? Let's use `collide`.
- Take a minute with your partner, and write down the contract for this function. You must have the entire contract written correctly to earn a point!
- *Countdown, assign points, and review.*
- So what is our Domain? A Number, representing the distance between rocket and asteroid! And our Range? The Number of seconds being counted down before they collide.
- `; collide : Number -> Number`
- So what comes next? Examples! Let's start with something easy - when the rocket and asteroid have just smashed into each other. How many miles apart are they then? Zero! And many seconds will it take for them to collide? Zero again, because they've already met!
- `; collide : distance -> time`
- `; If they start off d miles apart, how long will it take to collide?`
- `collide(0) = ...`
- So now we need to figure out what calculations go after the equals sign. Luckily, we have a hint: we know that whatever is it HAS to come out to zero, since we know that they've already collided when they are zero miles apart. If the rocket is going 80 miles/second, and the asteroid is going 70 miles/second, how fast are they approaching each other? Let students chew on this.
- 60 + 70 = 150, so we know that they are getting closer and closer together at a rate of 150 miles per second.
- This isn't that different from our `time` function: we have some speed, and we want to know how much time it will take to go a given distance. So what do we write for our example?
- `; collide : distance -> time`
- `; If they start off d miles apart, how long will it take to collide?`
- `collide(0) = 0/150`
- Let's do another example, to make sure we've got a handle on this thing. If they're traveling towards each other at 150 miles/second, how long will it take for them to meet if they start out 300 miles apart? (two seconds) How do you know? (300/150 = 2) Raise your hand if you can write this as an example. Have students do a few more examples...
- `; collide : distance -> time`
- `; If they start off d miles apart, how long will it take to collide?`
- `collide(0) = 0/150`
- `collide(300) = 300/150`
- `collide(5000) = 5,000/150`
- `collide(100,000) = 100,000/150`
- By now, some of you may have already figured out how to get the answer. If so, that's great! But suppose you still weren't sure - who can take us the rest of the way, and write the function? Take a volunteer...
- `; collide : distance -> time`
- `; If they start off d miles apart, how long will it take to collide?`
- `collide(0) = 0/150`

- `collide(300) = 300/150`
- `collide(5000) = 5,000/150`
- `collide(100,000) = 100,000/150`
- `collide(d) = d/150`
- Sometimes, the Design Recipe will get you to the answer without even having to finish. That's how powerful it is!
- Open your workbooks to Page 39. This page and the one after it are wildcards – you'll have until the end of class to finish them, and find out how many points you can get! Are you ready? On your marks, get set, GO!
- *TEACHERS: you can add as many pages as you like to the workbooks, using any algebra problem you like. We recommend using word problems from your students' algebra textbook, or even from your state's standardized test!*