Name: _____

Student Workbook

Fall, 2022 - Pyret Edition

Workbook v1.5

Brought to you by the Bootstrap team:

- Emmanuel Schanzer
- Emma Youndtsmith
- Kathi Fisler
- Shriram Krishnamurthi
- Dorai Sitaram
- Joe Politz
- Ben Lerner

Visual Designer: Colleen Murphy

Bootstrap is licensed under a Creative Commons 3.0 Unported License. Based on a work from www.BootstrapWorld.org. Permissions beyond the scope of this license may be available at contact@BootstrapWorld.org.

Computing Needs All Voices!

The pioneers pictured below are featured in our Computing Needs All Voices lesson. To learn more about them and their contributions, visit https://bit.ly/bootstrap-pioneers.

We are in the process of expanding our collection of pioneers. If there's someone else whose work inspires you, please let us know at https://bit.ly/pioneer-suggestion.

Notice and Wonder

Write down what you notice and wonder from the **What Most Schools Don't Teach** video.

"Notices" should be statements, not questions. What stood out to you? What do you remember? "Wonders" are questions.

What do you Notice?	What do you Wonder?

Windows and Mirrors

Think about the images and stories you've just encountered. Identify something(s) that served as a mirror for you, connecting you with your own identity and experience of the world. Write about who or what you connected with and why.

Identify something(s) from the film or the posters that served as a window for you, giving you insight into other people's experiences or expanding your thinking in some way.

Reflection: Problem Solving Advantages of Diverse Teams

This reflection is designed to follow reading LA Times Perspective: A solution to tech's lingering diversity problem? Try thinking about ketchup

1) The author argues that tech companies with diverse teams have an advantage. Why?

2) What suggestions did the article offer for tech companies looking to diversify their teams?

3) What is one thing of interest to you in the author's bio?

4) Think of a time when you had an idea that felt out of the box. Did you share your idea? Why or why not?

5) Can you think of a time when someone else had a strategy or idea that you would never have thought of, but was interesting to you and/or pushed your thinking to a new level?

6) Based on your experience of exceptions to mainstream assumptions, propose another pair of questions that could be used in place of "Where do you keep your ketchup?" and "What would you reach for instead?".

Introduction to Programming

The **Editor** is a software program we use to write Code. Our Editor allows us to experiment with Code on the right-hand side, in the **Interactions Area**. For Code that we want to *keep*, we can put it on the left-hand side in the **Definitions Area**. Clicking the "Run" button causes the computer to re-read everything in the Definitions Area and erase anything that was typed into the Interactions Area.

Data Types

Programming languages involve different ***data types***, such as Numbers, Strings, Booleans, and even Images.

- Numbers are values like `1`, `0.4`, `1/3`, and `-8261.003`.
  - Numbers are *usually* used for quantitative data and other values are *usually* used as categorical data.
  - In Pyret, any decimal *must* start with a 0. For example, `0.22` is valid, but `.22` is not.
- Strings are values like `"Emma"`, `"Rosanna"`, `"Jen and Ed"`, or even `"08/28/1980"`.
  - All strings *must* be surrounded in quotation marks.
- Booleans are either `true` or `false`.

All values evaluate to themselves. The program `42` will evaluate to `42`, the String `"Hello"` will evaluate to `"Hello"`, and the Boolean `false` will evaluate to `false`.

Operators

Operators (like `+`, `-`, `*`, `<`, etc.) work the same way in Pyret that they do in math.

- Operators are written between values, for example: `4 + 2`.
- In Pyret, operators must always have a space around them. `4 + 2` is valid, but `4+2` is not.
- If an expression has different operators, parentheses must be used to show order of operations. `4 + 2 + 6` and `4 + (2 * 6)` are valid, but `4 + 2 * 6` is not.

Applying Functions

Applying functions works much the way it does in math. Every function has a name, takes some inputs, and produces some output. The function name is written first, followed by a list of ***arguments*** in parentheses.

- In math this could look like $f(5)$ or $g(10, 4)$.
- In Pyret, these examples would be written as `f(5)` and `g(10, 4)`.
- Applying a function to make images would look like `star(50, "solid", "red")`.
- There are many other functions, for example `num-sqr`, `num-sqrt`, `triangle`, `square`, `string-repeat`, etc.

Functions have ***contracts***, which help explain how a function should be used. Every contract has three parts:

- The *Name* of the function - literally, what it's called.
- The *Domain* of the function - what *types of values* the function consumes, and in what order.
- The *Range* of the function - what *type of value* the function produces.

Numbers and Strings

Make sure you've loaded the code.pyret.org (CPO), clicked "Run", and are working in the ***Interactions Area***.

Numbers

1) Try typing `42` into the Interactions Area and hitting "Enter". What is the largest number the editor can handle?

2) Try typing `0.5`. Then try typing `.5`. Then try clicking on the answer. Experiment with other decimals. Explain what you understand about how decimals work in this programming language.

3) What happens if you try a fraction like `1/3`?

4) Try writing **negative** integers, fractions and decimals. What do you learn?

Strings

String values are always in quotes.

5) Is `42` the same as `"42"`? Why or why not? Write your answer below:

- Try typing your name *(in quotes!)*.
- Try typing a sentence like "I'm excited to learn to code!" *(in quotes!)*.
- Try typing your name with the opening quote, but *without the closing quote.* Read the error message!
- Now try typing your name *without any quotes.* Read the error message!

6) Explain what you understand about how strings work in this programming language.

Operators

7) Just like math, Pyret has ***operators*** like `+`, `-`, `*` and `/`. Try typing in `4 + 2`, and then `4+2` (without the spaces). What can you conclude from this?

8) Type in the following expressions, **one at a time**: `4 + 2 * 6`, `(4 + 2) * 6`, `4 + (2 * 6)`. What do you notice?

9) Try typing in `4 + "cat"`, and then `"dog" + "cat"`. What can you conclude from this?

Booleans

Boolean-producing expressions are yes-or-no questions and will always evaluate to either `true` ("yes") or `false` ("no"). What will each of the expressions below evaluate to? *Write down your prediction in the blanks provided and then type the code into the Interactions Area to see what it returns.*

	Prediction	Result		Prediction	Result
1) `3 <= 4`			2) `"a" > "b"`		
3) `3 == 2`			4) `"a" < "b"`		
5) `2 < 4`			6) `"a" == "b"`		
7) `5 >= 5`			8) `"a" <> "a"`		
9) `4 >= 6`			10) `"a" >= "a"`		
11) `3 <> 3`			12) `"a" <> "b"`		

13) In your own words, describe what `<` does.

14) In your own words, describe what `>=` does.

15) In your own words, describe what `<>` does.

	Prediction:	Result:
16) `string-contains("catnap", "cat")`		
17) `string-contains("cat", "catnap")`		

18) How many **Numbers** are there in the entire universe?

19) How many **Strings** are there in the entire universe?

20) How many **Booleans** are there in the entire universe?

Applying Functions

Type this line of code into the Interactions Area and hit "Enter":

```
triangle(50, "solid", "red")
```

1) What is the name of this function? _____

2) What did the expression evaluate to? _____

3) How many arguments does `triangle` expect? _____

4) What data type does the `triangle` function produce? _____

Catching Bugs

The following lines of code are all BUGGY! Read the code and the error messages to identify the mistake.

5) `triangle(20, "solid" "red")`

```
Pyret didn't understand your program around
triangle(20, "solid" "red")
```

Can you spot the mistake? _____

6) `triangle(20, "solid")`

```
This application expression errored:
triangle(20, "solid")
2 arguments were passed to the operator. The operator evaluated to a function accepting 3
parameters. An application expression expects the number of parameters and arguments to be the
same.
```

Can you spot the mistake? _____

7) `triangle(20, 10, "solid", "red")`

```
This application expression errored:
triangle(20, 10, "solid", "red")`
4 arguments were passed to the operator. The operator evaluated to a function accepting 3
parameters. An application expression expects the number of parameters and arguments to be the
same.
```

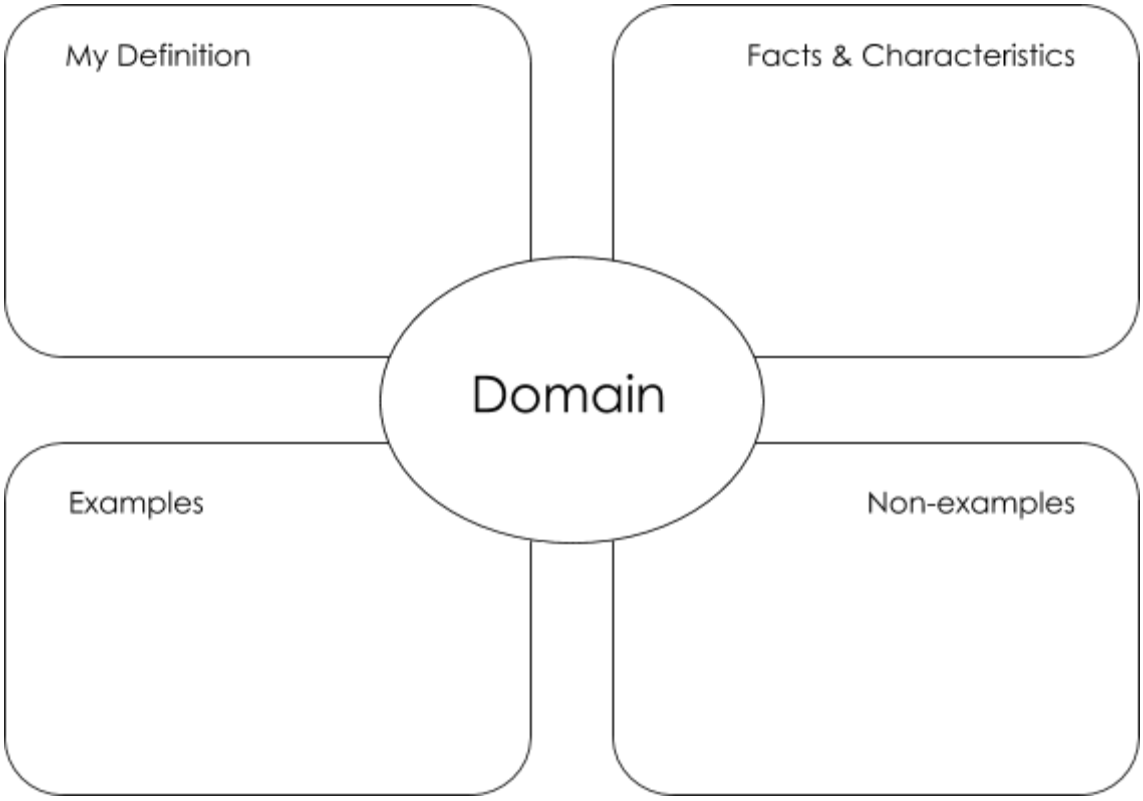
Can you spot the mistake? _____

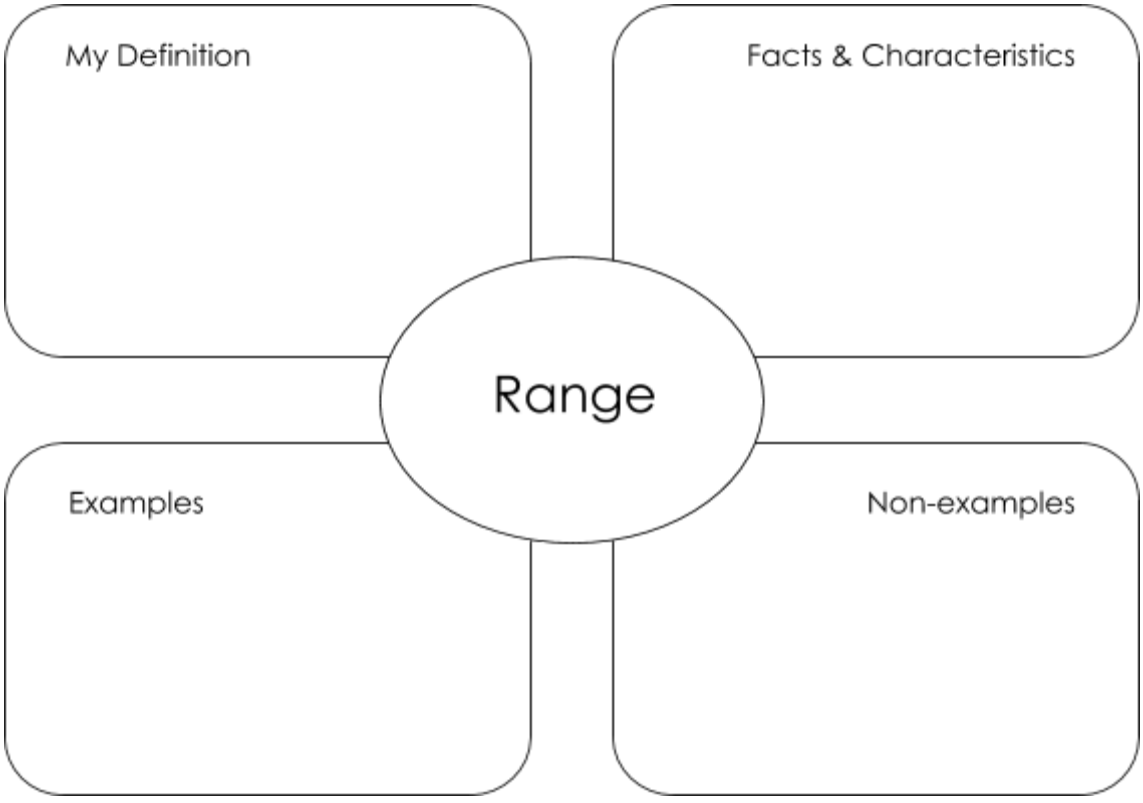
8) `triangle (20, "solid", "red")`

```
Pyret thinks this code is probably a function call:
triangle (20, "solid", "red")
Function calls must not have space between the function expression and the arguments.
```

Can you spot the mistake? _____

Domain and Range





Practicing Contracts: Domain & Range

Consider the following contract:

```
is-beach-weather :: Number, String -> Boolean
```

1) What is the **Name** of this function? _____

2) How many arguments are in this function's **Domain**? _____

3) What is the **Type** of this function's **first argument**? _____

4) What is the **Type** of this function's **second argument**? _____

5) What is the **Range** of this function? _____

6) Circle the expression below that shows the correct application of this function, based on its contract.

A. `is-beach-weather(70, 90)`

B. `is-beach-weather(80, 100, "cloudy")`

C. `is-beach-weather("sunny", 90)`

D. `is-beach-weather(90, "stormy weather")`

Consider the following contract:

```
cylinder :: Number, Number, String -> Image
```

7) What is the **Name** of this function? _____

8) How many arguments are in this function's **Domain**? _____

9) What is the **Type** of this function's **first argument**? _____

10) What is the **Type** of this function's **second argument**? _____

11) What is the **Type** of this function's **third argument**? _____

12) What is the **Range** of this function? _____

13) Circle the expression below that shows the correct application of this function, based on its contract.

A. `cylinder("red", 10, 60)`

B. `cylinder(30, "green")`

C. `cylinder(10, 25, "blue")`

D. `cylinder(14, "orange", 25)`

Matching Expressions and Contracts

Match the contract (left) with the expression described by the function being used (right).

Contract		Expression
`# make-id :: String, Number -> Image`	1	A `make-id("Savannah", "Lopez", 32)`
`# make-id :: String, Number, String -> Image`	2	B `make-id("Pilar", 17)`
`# make-id :: String -> Image`	3	C `make-id("Akemi", 39, "red")`
`# make-id :: String, String -> Image`	4	D `make-id("Raïssa", "McCracken")`
`# make-id :: String, String, Number -> Image`	5	E `make-id("von Einsiedel")`

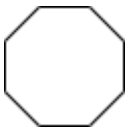
Contract		Expression
`# is-capital :: String, String -> Boolean`	6	A `show-pop("Juneau", "AK", 31848)`
`# is-capital :: String, String, String -> Boolean`	7	B `show-pop("San Juan", 395426)`
`# show-pop :: String, Number -> Image`	8	C `is-capital("Accra", "Ghana")`
`# show-pop :: String, String, Number -> Image`	9	D `show-pop(3751351, "Oklahoma")`
`# show-pop :: Number, String -> Number`	10	E `is-capital("Albany", "NY", "USA")`

Using Contracts

`ellipse:: Number, Number, String, String -> Image`

	Use the contract to write an expression that generates a similar image:
	Use the contract to write an expression that generates a similar image:
What changes with the first Number?	
What about the shape changes with the second Number?	
Write an expression using `ellipse` to produce a circle.	

`regular-polygon:: Number, Number, String, String -> Image`

	Use the contract to write an expression that generates a similar image:
	Use the contract to write an expression that generates a similar image:
What changes with the first Number?	
What about the shape changes with the second Number?	
Use `regular-polygon` to write an expression for a square!	
How would you describe a **regular polygon** to a friend?	

Triangle Contracts

1) What kind of triangle does the `triangle` function produce? _____

There are lots of other kinds of triangles! And Pyret has lots of other functions that make triangles!

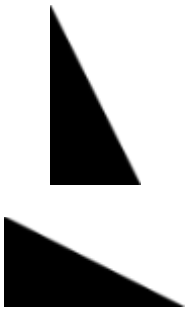
```
triangle :: (size:: Number, style :: String, color :: String) -> Image
right-triangle :: (base::Number, height::Number, style::String, color::String) -> Image
isosceles-triangle :: (leg::Number, angle::Number, style::String, color::String) -> Image
```

2) Why do you think `triangle` only needs one number, while `right-triangle` and `isosceles-triangle` need two numbers and `triangle-sas` needs three?

_____

_____

3) Write `right-triangle` expressions for the images below. *One argument for each should be* `100`.



_____

_____

4) What do you think the numbers in `right-triangle` represent? _____

5) Write `isosceles-triangle` expressions for the images below. *1 argument for each should be* `100`.

_____

_____

6) What do you think the numbers in `isosceles-triangle` represent?

_____

7) Write 2 expressions that would build **right-isosceles** triangles. Use `right-triangle` for one expression and `isosceles-triangle` for the other expression.

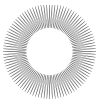
_____

_____

Radial Star

```
radial-star :: (
    points :: Number,
    inner-radius :: Number,
    full-radius :: Number,
    style :: String,
    color :: String
) -> Image
```

Using the detailed contract above, match each image to the expression that describes it.

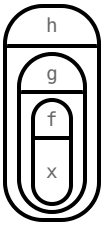
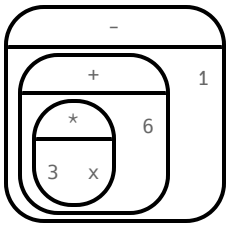
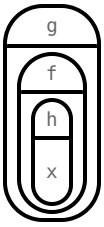
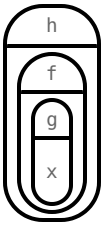
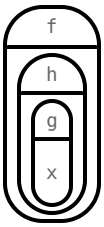
Image			Expression
	1	A	`radial-star(5, 50, 200, "solid", "black")`
	2	B	`radial-star(7, 100, 200, "solid", "black")`
	3	C	`radial-star(7, 100, 200, "outline", "black")`
	4	D	`radial-star(10, 150, 200, "solid", "black")`
	5	E	`radial-star(10, 20, 200, "solid", "black")`
	6	F	`radial-star(100, 20, 200, "outline", "black")`
	7	G	`radial-star(100, 100, 200, "outline", "black")`

What's on your mind?

Diagramming Function Composition

`f :: Number -> Number` `Consumes a number, multiplies by 3 to produce the result`	`g :: Number -> Number` `Consumes a number, adds six to produce the result`	`h :: Number -> Number` `Consumes a number, subtracts one to produce the result`
$f(x) = 3x$	$g(x) = x + 6$	$h(x) = x - 1$

For each function composition diagrammed below, translate it into the equivalent Circle of Evaluation for Order of Operations. Then write expressions for *both* versions of the Circles of Evaluation, and evaluate them for $x = 4$. The first one has been completed for you.

Function Composition	Order of Operations	Translate & Evaluate	
1) 		Composition:	`h(g(f(x)))`
		Operations:	`((3 * x) + 6) - 1`
		Evaluate for x = 4	$h(g(f(4))) = 17$
2) 		Composition:	
		Operations:	
		Evaluate for x = 4	
3) 		Composition:	
		Operations:	
		Evaluate for x = 4	
4) 		Composition:	
		Operations:	
		Evaluate for x = 4	

Function Composition — Green Star

1) Draw a Circle of Evaluation and write the Code for a **solid, green star, size 50**.

Circle of Evaluation:

Code: _____

Using the star described above as the **original**, draw the Circles of Evaluation and write the Code for each exercise below.

2) A solid, green star, that is triple the size of the original (using `scale`)	3) A solid, green star, that is half the size of the original (using `scale`)
4) A solid, green star of size 50 that has been rotated 45 degrees counter-clockwise	5) A solid, green star that is 3 times the size of the original **and** has been rotated 45 degrees

Function Composition — Your Name

You'll be investigating these functions with your partner:

```
# text :: String, Number, String -> Image
# flip-horizontal :: Image -> Image
# flip-vertical :: Image -> Image
# frame :: Image -> Image
# above :: Image, Image -> Image
# beside :: Image, Image -> Image
```

1) In the editor, write the code to make an image of your name in big letters in a color of your choosing using `text`. Then draw the Circle of Evaluation and write the Code that will create the image.

Circle of Evaluation:

Code:

Using the "image of your name" described above as the **original**, draw the Circles of Evaluation and write the Code for each exercise below. Test your ideas in the editor to make sure they work.

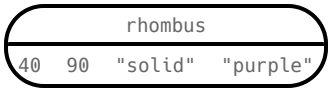
2) The framed "image of your name".	3) The "image of your name" flipped vertically.
4) The "image of your name" above "the image of your name" flipped vertically.	5) The "image of your name" flipped horizontally beside "the image of your name".

Function Composition — scale-xy

You'll be investigating these two functions with your partner:

```
# scale-xy :: Number, Number, Image -> Image
```

```
# overlay :: Image, Image -> Image
```

The Image:	Circle of Evaluation:	Code:
		`rhombus(40, 90, "solid", "purple")`

Starting with the image described above, write the Circles of Evaluation and Code for each exercise below. Be sure to test your code in the editor!

1) A purple rhombus that is stretched 4 times as wide.	2) A purple rhombus that is stretched 4 times as tall
3) The tall rhombus from #1 overlayed on the wide rhombus (#2).	★ Overlay a red rhombus onto the last image you made in #3.

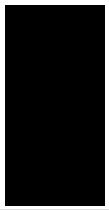
More than one way to Compose an Image!

What image will each of the four expressions below evaluate to? If you're not sure, type them into the Interactions Area and see if you can figure out how the code constructs its image.

```
beside(rectangle(200, 100, "solid", "black"), square(100, "solid", "black"))

scale-xy(1, 2, square(100, "solid", "black"))

scale(2, rectangle(100, 100, "solid", "black"))

above(
    rectangle(100, 50, "solid", "black"),
    above(
        rectangle(200, 100,  "solid", "black"),
        rectangle(100, 50, "solid", "black")))
```

For each image below, identify 2 expressions that could be used to compose it. *The bank of expressions at the top of the page includes one possible option for each image.*

1



2

3

★

20

Defining Values

In math, we use **values** like $-98.1, 2/3$ and 42. In math, we also use **expressions** like 1×3, $\sqrt{16}$, and $5 - 2$. These evaluate to results, and typing any of them in as code produces some answer.

Math also has **definitions**. These are different from values and expressions, because *they do not produce results*. Instead, they simply create names for values, so that those names can be re-used to make the Math simpler and more efficient.

Definitions always have both a name and an expression. The name goes on the left and the value-producing expression goes on the right, separated by an equals sign:

$x = 4$

$y = 9 + x$

The name is defined to be the result of evaluating the expression. Using the above examples, we get "x is defined to be 4, and y is defined to be 13". **Important: there is no "answer" to a definition**, and typing in a definition as code will produce no result.

Notice that *definitions can refer to previous definitions*. In the example above, the definition of `y` refers to `x`. But `x`, on the other hand, *cannot* refer to `y`. Once a value has been defined, it can be used in later expressions.

In Pyret, these definitions are written the *exact same way*:

Try typing these definitions into the Definitions Area on the left, clicking "Run", and then *using* them in the Interactions Area on the right.

```
x = 4
y = 9 + x
```

Just like in math, definitions in our programming language can only refer to previously-defined values.

Here are a few more value definitions. Feel free to type them in, and make sure you understand them.

```
x = 5 + 1
y = x * 7
food = "Pizza!"
dot = circle(y, "solid", "red")
```

Defining Values - Explore

Open the Defining Values Starter File and click "Run".

1) What do you Notice?

2) What do you Wonder?

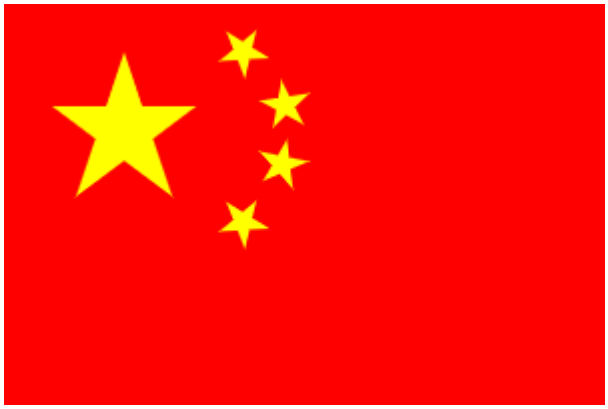
Look at the expressions listed below. Think about what you expect each of them to produce. Then, test them out one at a time in the Interactions Area.

- `x`
- `x + 5`
- `y - 9`
- `x * y`
- `z`
- `t`
- `gold-star`
- `my-name`
- `swamp`
- `c`

3) What have you learned about defining values?

4) Define at least 2 more variables in the Definitions Area, click "Run" and test them out. Once you know they're working, record the code you used below.

Defining Values - Chinese Flag



1) What image do you see repeated in the flag? _____

2) In the code below, highlight or circle all instances of the expression that makes the repeated image.

```
china =
  put-image(
    rotate(40,star(15,"solid","yellow")),
    120, 175,
    put-image(
      rotate(80,star(15,"solid","yellow")),
      140, 150,
      put-image(
        rotate(60,star(15,"solid","yellow")),
        140, 120,
        put-image(
          rotate(40,star(15,"solid","yellow")),
          120, 90,
          put-image(scale(3,star(15,"solid","yellow")),
            60, 140,
            rectangle(300, 200, "solid", "red"))))))
```

3) Write the code to define a value for the repeated expression.

_____

4) Open the Chinese Flag Starter File and click "Run".

- Type `china` into the Interactions Area and click **Enter**.
- **Save a copy** of the file, and simplify the flag code using the value you defined.
- Click "Run", and confirm that you still get the same image as the original.
- Now change the color of all of the stars to black, in both files.
- Then change the size of the stars.

5) Why is it helpful to define values for repeated images?

_____

_____

Challenge:

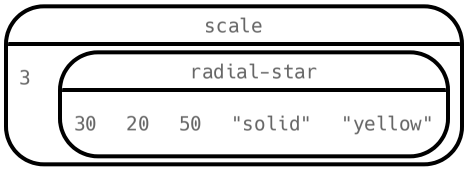
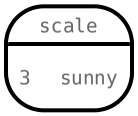
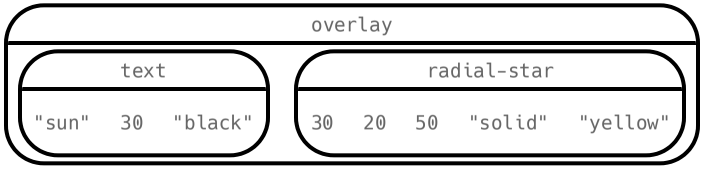
- This file uses a function we haven't seen before! What is it? _____
- Can you figure out its contract? *Hint: Focus on the last instance of the function.*

_____

Why Define Values?

1) Complete the table using the first row as an example.

2) Write the code to define the value of `sunny`. _____

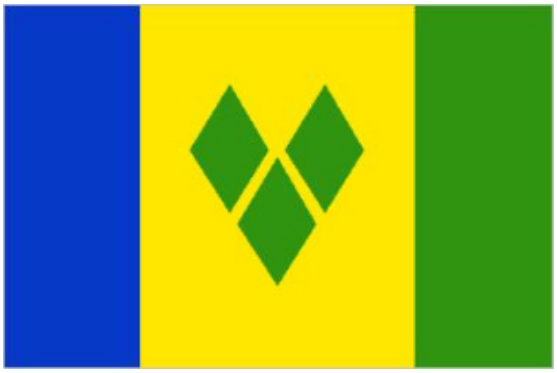
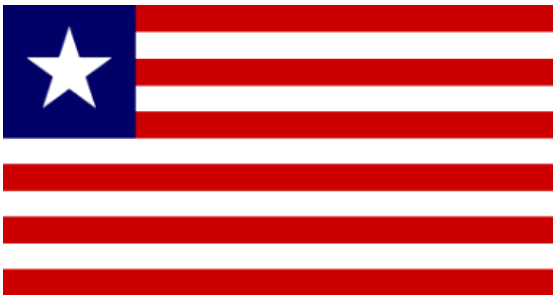
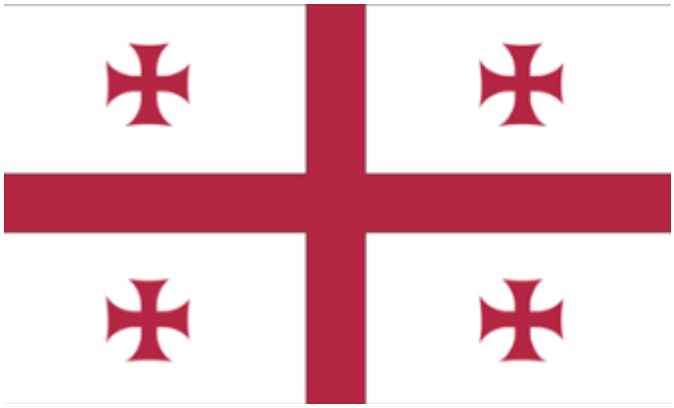
Original Circle of Evaluation & Code	→	Use the *defined value* `sunny` to simplify!
	→	
Code: `scale(3, radial-star(30, 20, 50, "solid", "yellow"))`	→	Code: `scale(3, sunny)`
	→	
Code: `frame(radial-star(30, 20, 50, "solid", "yellow"))`	→	Code:
	→	
Code: `overlay(text("sun", 30, "black"), radial-star(30, 20, 50, "solid", "yellow"))`	→	Code:

3) Test your code in the editor and make sure it produces what you would expect it to.

Which Value(s) Would it Make Sense to Define?

For each of the images below, identify which element(s) you would want to define before writing code to compose the image.

Hint: what gets repeated?

Philippines	St. Vincent & the Grenadines
	
_____	_____
Liberia	**Republic of Georgia**
	
_____	_____
Quebec	**South Korea**
	
_____	_____

Writing Code using Defined Values

1) On the line below, **write the Code** to define `PRIZE-STAR` as a pink, outline star of size 65.

Using the `PRIZE-STAR` definition from above, draw the Circle of Evaluation and write the Code for each of the exercises. One Circle of Evaluation has been done for you.

2) The outline of a pink star that is three times the size of the original (using `scale`) **Circle of Evaluation:** 	3) The outline of a pink star that is half the size of the original (using `scale`) **Circle of Evaluation:**
Code:	**Code:**
4) The outline of a pink star that is rotated 45 degrees *(It should be the same size as the original.)* **Circle of Evaluation:**	5) The outline of a pink star that is three times as big as the original **and** has been rotated 45 degrees **Circle of Evaluation:**
Code:	**Code:**

6) How does defining values help you as a programmer?

Estimating Coordinates

```
dot = circle(50, "solid", "red")
background = rectangle(300, 200, "outline", "black")
```

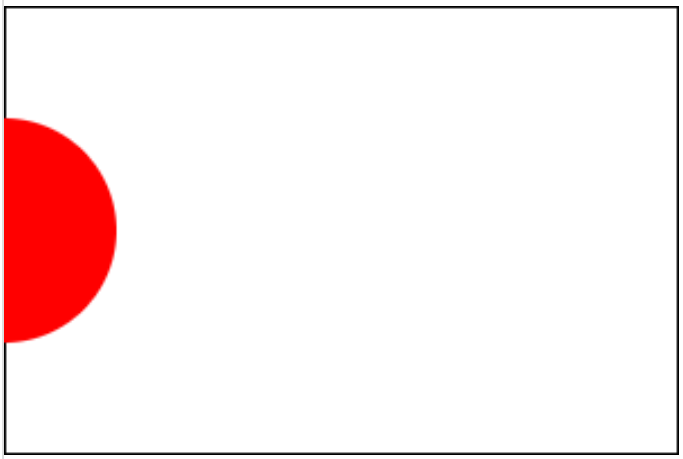
Think of the background image as a sheet of graph paper with the origin (0,0) in the bottom left corner. The width of the rectangle is 300 and the height is 200. The numbers in `put-image` specify a point on that graph paper, where the center of the top image (in this case `dot`) should be placed.

Estimate: What coordinates for the `dot` created each of the following images?

A

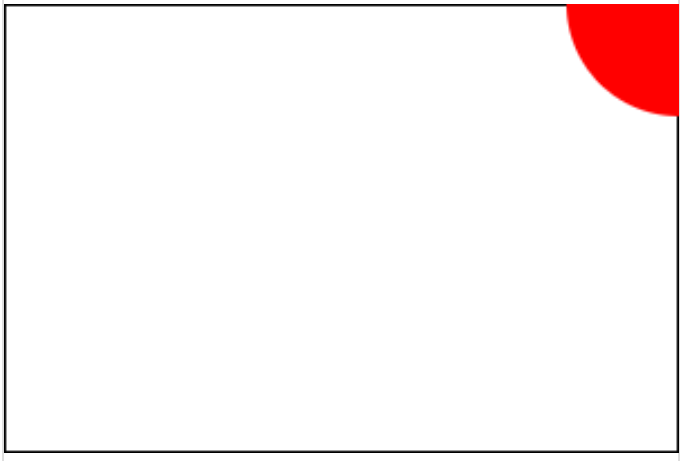
put-image(dot, _____, _____background)

B



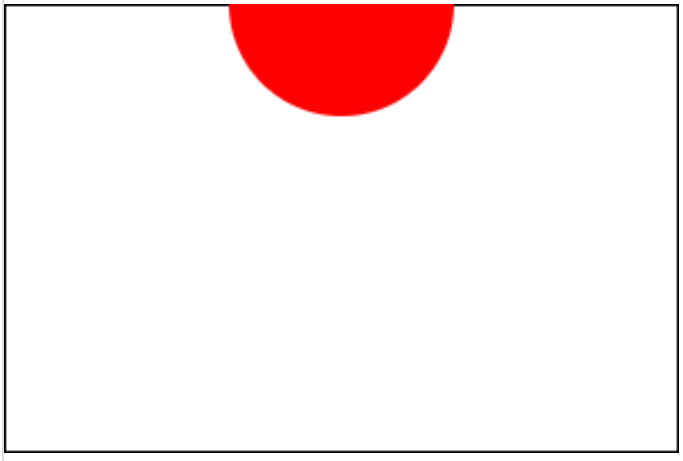
put-image(dot, _____, _____background)

C



put-image(dot, _____, _____background)

D



put-image(dot, _____, _____background)

Decomposing Flags

Each of the flags below is shown with their width and height. Identify the shapes that make up each flag. Use the flag's dimensions to estimate the dimensions of the different shapes. Then estimate the x and y coordinates for the point at which the center of each shape should be located on the flag. *Hint: The bottom left corner of each flag is at (0,0) and the top right corner is given by the flags dimensions.*

Cameroon (450 x 300)

shape:	color:	width:	height:	x	y

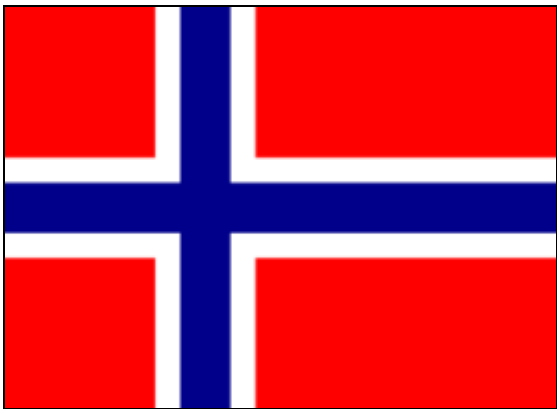
Chile (420 x 280)

shape:	color:	width:	height:	x	y

Panama (300 x 200)

shape:	color:	width:	height:	x	y

Norway (330 x 240)



shape:	color:	width:	height:	x	y

Solving Word Problems

Being able to see functions as Contracts, Examples or Definitions is like having three powerful tools. These representations can be used together to solve word problems!

1) When reading a word problem, the first step is to figure out the **Contract** for the function you want to build. Remember, a Contract must include the Name, Domain and Range for the function!

2) Then we write a **Purpose Statement**, which is a short note that tells us what the function *should do* . Professional programmers work hard to write good purpose statements, so that other people can understand the code they wrote!

3) Next, we write at least two **Examples**. These are lines of code that show what the function should do for a *specific* input. Once we see examples of at least two inputs, we can *find a pattern* and see which parts are changing and which parts aren't.

4) To finish the Examples, we circle the parts that are changing, and label them with a short **variable name** that explains what they do.

5) Finally, we **define the function** itself! This is pretty easy after you have some examples to work from: we copy everything that didn't change, and replace the changeable stuff with the variable name!

Matching Word Problems and Purpose Statements

Match each word problem below to its corresponding purpose statement.

Annie got a new dog, Xavier, that eats about 5 times as much as her little dog, Rex, who is 10 years old. She hasn't gotten used to buying enough dogfood for the household yet. Write a function that generates an estimate for how many pounds of food Xavier will eat, given the amount of food that Rex usually consumes in the same amount of time.	1		A	Consume the pounds of food Rex eats and add 5.
Adrienne's raccoon, Rex, eats 5 more pounds of food each week than her pet squirrel, Lili, who is 7 years older. Write a function to determine how much Lili eats in a week, given how much Rex eats.	2		B	Consume the pounds of food Rex eats and subtract 5.
Alejandro's rabbit, Rex, poops about 1/5 of what it eats. His rabbit hutch is 10 cubic feet. Write a function to figure out how much rabbit poop Alejandro will have to clean up depending on how much Rex has eaten.	3		C	Consume the pounds of food Rex eats and multiply by 5.
Max's turtle, Rex, eats 5 pounds less per week than his turtle, Harry, who is 2 inches taller. Write a function to calculate how much food Harry eats, given the weight of Rex's food.	4		D	Consume the pounds of food Rex eats and divide by 5.

Writing Examples from Purpose Statements

We've provided contracts and purpose statements to describe two different functions. Write examples for each of those functions.

Contract and Purpose Statement

Every contract has three parts…

triple:: *Number* -> *Number*

function name | *Domain* | *Range*

Consumes a Number and triples it.

what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____

function name | *input(s)* | *what the function produces*

_____ (_____) **is** _____

function name | *input(s)* | *what the function produces*

end

Contract and Purpose Statement

Every contract has three parts…

upside-down:: *Image* -> *Image*

function name | *Domain* | *Range*

Consumes an image, and turns it upside down by rotating it 180 degrees.

what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is**

function name | *input(s)*

_____

what the function produces

_____ (_____) **is** _____

function name | *input(s)* | *what the function produces*

end

Fixing Purpose Statements

Beneath each of the word problems below is a purpose statement that is either missing information or includes unnecessary information. Write an improved version of each purpose statement beneath the original.

1) **Word Problem:** *The New York City ferry costs $2.75 per ride. The Earth School requires two chaperones for any field trip. Write a function* `fare` *that takes in the number of students in the class and returns the total fare for the students and chaperones.*

Purpose Statement: Define a function `fare` to take in the number of students and add 2.

Improved Purpose Statement:

2) **Word Problem:** *It is tradition for the Green Machines to go to Humpy Dumpty's for ice cream with their families after their soccer games. Write a function* `cones` *to take in the number of kids and calculate the total bill for the team, assuming that each kid brings two family members and cones cost $1.25.*

Purpose Statement: Define a function `cones` to take in the number of kids on the team and multiply it by 1.25.

Improved Purpose Statement:

3) **Word Problem:** *The cost of renting an ebike is $3 plus an additional $0.12 per minute. Write a function* `ebike` *that will calculate the cost of a ride, given the number of minutes ridden.*

Purpose Statement: Define a function `ebike` to take in the number of minutes and multiply it by 3.12.

Improved Purpose Statement:

4) **Word Problem:** *Suleika is a skilled house painter at only age 21. She has painted hundreds of rooms and can paint about 175 square feet an hour. Write a function* `paint` *that takes in the number of square feet of the job and calculates how many hours it will take her.*

Purpose Statement: Define a function `paint` to take in the number of square feet of walls in a house and divide them by 175 to calculate the number of hours that it will take 21 year-old Suleika to complete the paint job.

Improved Purpose Statement:

Word Problem: rocket-height

Directions: A rocket blasts off, and is now traveling at a constant velocity of 7 meters per second. Use the Design Recipe to write a function `rocket-height`, which takes in a number of seconds and calculates the height.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Intro to Data Structures

Word Problem: double-radius

Directions: Write a function *double-radius* , which takes in a radius and a color. It produces an outlined circle of whatever color was passed in, whose radius is twice as big as the input.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Word Problem: double-width

Directions: Write a function *double-width* , which takes in a number (the length of a rectangle) and produces a rectangle whose length is twice the given length.

Contract and Purpose Statement

Every contract has three parts…

```
#  _____  ::  _____  ->  _____
```
function name *Domain* *Range*

```
#  _____
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

```
examples:
  _____ ( _____ ) is _____
  _____ ( _____ ) is _____
end
```
function name *input(s)* *what the function produces*

function name *input(s)* *what the function produces*

Definition

Write the definition, giving variable names to all your input values…

```
fun _____ ( _____ ):
  _____
end
```
function name *variable(s)*

what the function does with those variable(s)

Word Problem: next-position

Directions: Write a function *next-position* , which takes in two numbers (an x- and y-coordinate) and returns a DeliveryState, increasing the x-coordinate by 5 and decreasing the y-coordinate by 5.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Data Structure: CakeType

```
# A CakeType is a flavor, layers, & is-iceCream
data CakeType:
    | cake(_____________________________________________________
           _____________________________________________________
           _____________________________________________________)
      end
```

To make an instance of this structure, I would write:

```
cake1 = _____________________________________________________

cake2 = _____________________________________________________
```

To access the fields of `cake2`, I would write:

```
        _____________________________________________________

        _____________________________________________________

        _____________________________________________________
```

Word Problem: taller-than

Directions: Write a function called *taller-than* , which consumes two CakeTypes, and produces true if the number of layers in the first CakeType is greater than the number of layers in the second.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Word Problem: will-melt

Directions: Write a function called *will-melt* , which takes in a CakeType and a temperature, and returns true if the temperature is greater than 32 degrees, AND the CakeType is an ice-cream cake.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Vocabulary Practice

Below is a new structure definition:

```
data MediaType:
  | book(
      title :: String,
      author :: String,
      pubyear :: Number)
end

# an example book:
book1 = book("1984", "Orwell", 1949)
```

Fill in the blanks below with the vocabulary term that applies to each name. Here are the terms to choose from:

contract	example
header	field
data type	instance
constructor	data block
name	purpose

`author` is a _____

`book` is a _____

`MediaType` is a _____

`book1` is a _____

`title` is a _____

`data` … **`end`** is a _____

Unit 3 (Structures, Reactor, & Animations)

Identifying Animation Data Worksheet

Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	Data Type (Number, String, Image, Boolean ...)

Design a Data Structure

```
# a _________________State is ____________________________________

data _________________State:

    | ____________( ____________________________________________________

                    ____________________________________________________

                    ____________________________________________________

end
```

Make a sample instance for each sketch from the previous page:

```
sketchA = ____________________________________________________

sketchB = ____________________________________________________

sketchC = ____________________________________________________
```

Word Problem: draw-state

Write a function called *draw-state* , which takes in a SunsetState and returns an image in which the sun (a circle) appears at the position given in the SunsetState. The sun should be behing the horizon (the ground) once it is low in the sky.

```
{}draw-state :: _____ -> Image
{}
{}# _____
```

SUN =	
GROUND =	
SKY =	

```
fun _____(_____):
{}    _____
{}    _____
{}    _____
{}end
```

Word Problem: next-state-tick

Directions: Write a function called *next-state-tick* , which takes in a SunsetState and returns a SunsetState in which the new x-coordinate is 8 pixels larger than in the given SunsetState and the y-coordinate is 4 pixels smaller than in the given SunsetState.

Contract and Purpose Statement

Every contract has three parts…

```
#  _____  ::  _____  ->  _____
```
function name *Domain* *Range*

```
#  _____
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

```
examples:
  _____ ( _____ ) is _____
  _____ ( _____ ) is _____
end
```
function name *input(s)* *what the function produces*

function name *input(s)* *what the function produces*

Definition

Write the definition, giving variable names to all your input values…

```
fun _____ ( _____ ):
  _____
end
```
function name *variable(s)*

what the function does with those variable(s)

Identifying Animation Data Worksheet

Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	Data Type (Number, String, Image, Boolean ...)

Design a Data Structure

```
# a _________________State is ____________________________________

data _________________State:

    | ____________( ____________________________________________________

                    ____________________________________________________

                    ____________________________________________________

end
```

Make a sample instance for each sketch from the previous page:

```
_____sketchA_____ = ____________________________________________________

_____sketchB_____ = ____________________________________________________

_____sketchC_____ = ____________________________________________________
```

Identifying Animation Data Worksheet

Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	Data Type (Number, String, Image, Boolean ...)

Design a Data Structure

```
# a _________________State is ____________________________________

data _________________State:

    | ____________( ____________________________________________________

                    ____________________________________________________

                    ____________________________________________________

end
```

Make a sample instance for each sketch from the previous page:

```
sketchA = ____________________________________________________

sketchB = ____________________________________________________

sketchC = ____________________________________________________
```

Identifying Animation Data Worksheet

Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	Data Type (Number, String, Image, Boolean ...)

Design a Data Structure

```
# a _________________State is ____________________________________

data _________________State:

    | ____________( ____________________________________________________

                    ____________________________________________________

                    ____________________________________________________

end
```

Make a sample instance for each sketch from the previous page:

```
_____sketchA_____ = ____________________________________________________

_____sketchB_____ = ____________________________________________________

_____sketchC_____ = ____________________________________________________
```

Identifying Animation Data Worksheet

Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	Data Type (Number, String, Image, Boolean ...)

Design a Data Structure

```
# a _________________State is ____________________________________

data _________________State:

    | ____________( ____________________________________________________

                    ____________________________________________________

                    ____________________________________________________

end
```

Make a sample instance for each sketch from the previous page:

```
sketchA = ____________________________________________________

sketchB = ____________________________________________________

sketchC = ____________________________________________________
```

Functions That Ask Questions

Word Problem: location

Directions: Write a function called location, which consumes a DeliveryState, and produces a String representing the location of a box: either "road", "delivery zone", "house", or "air".

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Syntax and Style Bug Hunting: Piecewise Edition

	Buggy Code	Correct Code / Explanation

1	`fun piecewisefun(n):`
	` if (n > 0): n`
	` else: 0`	

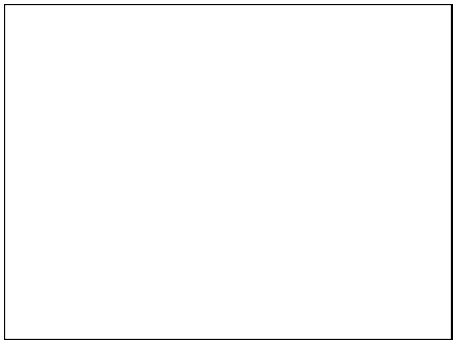
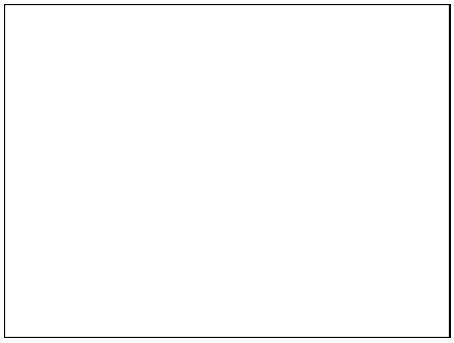
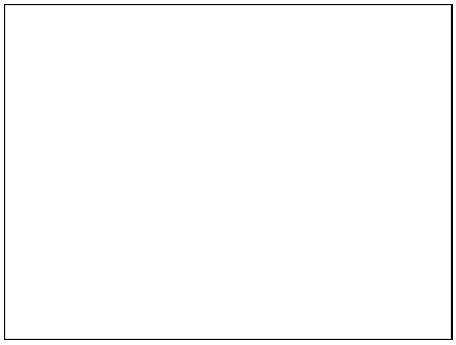
2	`fun cost(topping):`
	` if string-equal(topping,`
	`"pepperoni"): 10.50`
	` else string-equal(topping,`
	`"cheese"): 9.00`
	` else string-equal(topping,`
	`"chicken"): 11.25`
	` else string-equal(topping,`
	`"broccoli"): 10.25`
	` else: "That's not on the menu!"`
	` end`
	`end`	

3	`fun absolute-value(a b):`
	` if a > b: a - b`
	` b - a`
	` end`
	`end`	

4	`fun best-function(f):`
	` if string-equal(f, “blue”):`
	` “you win!”`
	` else if string-equal(f, “blue”):`
	` “you lose!”`
	` else if string-equal(f, “red”):`
	` “Try again!”`
	` else: “Invalid entry!”`
	` end`
	`end`	

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG …)	data type (Number, String, Image, Boolean …)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

1) Make a sample instance for each sketch from the previous page:

_____ =

_____ =

_____ =

2) Write at least one NEW example for one of the functions on your To-Do list

3) If you have another function on your To-Do list, write at least one NEW example

Word Problem: draw-sun

Directions: Write a function called draw-sun, which consumes a SunsetState, and produces an image of a sun (a solid, 25 pixel circle), whose color is "yellow", when the sun's y-coordinate is greater than 225, "orange", when its y-coordinate is between 150 and 225, and "red" otherwise.

Contract and Purpose Statement

Every contract has three parts…

```
#  _____  ::  _____  ->  _____
```
function name *Domain* *Range*

```
#  _____
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

```
examples:
  _____ ( _____ ) is _____
  _____ ( _____ ) is _____
  _____ ( _____ ) is _____
  _____ ( _____ ) is _____
end
```
function name *input(s)* *what the function produces*

Definition

Write the definition, giving variable names to all your input values…

```
fun _____ ( _____ ):
  _____
end
```
function name *variable(s)* *what the function does with those variable(s)*

Key Events

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	data type (Number, String, Image, Boolean ...)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐
next-state-tick	*If the Data Structure changed, or the animation happens automatically*	☐	☐
next-state-key	*If the Data Structure changed, or a keypress triggers the animation*	☐	☐
reactor	*If either next-state function is new*	☐	☐

1) Make a sample instance for each sketch from the previous page:

```
FULLPET = pet(100, 100)

MIDPET = pet(50, 75)

LOSEPET = pet(0, 0)
```

2) Write at least one NEW example for one of the functions on your To-Do list

next-state-tick(FULLPET) is pet(FULLPET.hunger – 2, FULLPET.sleep – 1)

next-state-tick(MIDPET) is pet(MIDPET.hunger – 2, MIDPET.sleep – 1)

next-state-tick(LOSEPET) is LOSEPET

3) If you have another function on your To-Do list, write at least one NEW example

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG ...)	data type (Number, String, Image, Boolean ...)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

1) Make a sample instance for each sketch from the previous page:

_____ =

_____ =

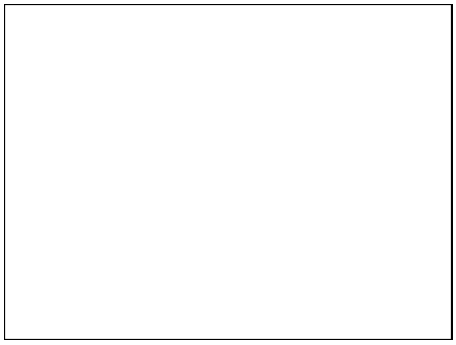
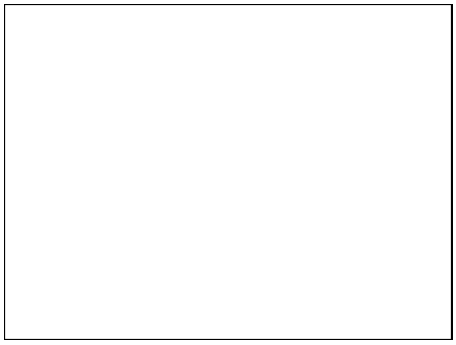
_____ =

2) Write at least one NEW example for one of the functions on your To-Do list

3) If you have another function on your To-Do list, write at least one NEW example

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG …)	data type (Number, String, Image, Boolean …)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

1) Make a sample instance for each sketch from the previous page:

_____ =

_____ =

_____ =

2) Write at least one NEW example for one of the functions on your To-Do list

3) If you have another function on your To-Do list, write at least one NEW example

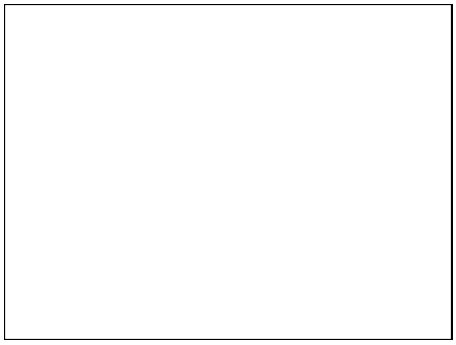
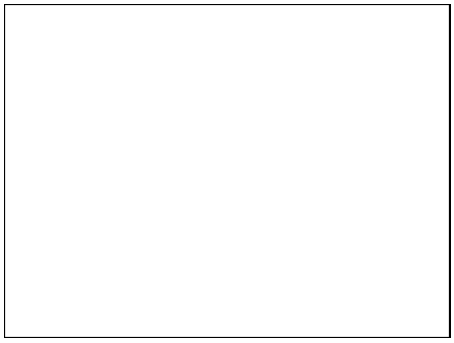
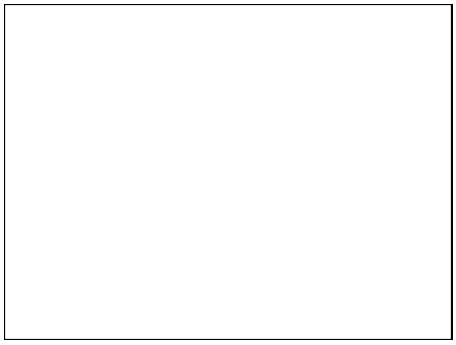
Refactoring

Your Own Drawing Functions

Build Your Own Animation

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG …)	data type (Number, String, Image, Boolean …)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

```
# a _____State is _____________________________________
{}data _____State:
{}| _____ (_____________________________________________
{}          _____________________________________________
{}          _____________________________________________
{}          _____________________________________________)
{}end

_____ = _____________________________________________
{}_____ = _____________________________________________
{}_____ = _____________________________________________

_____________________________________________________
{}_____________________________________________________
{}_____________________________________________________
{}_____________________________________________________
{}_____________________________________________________
```

Collisions

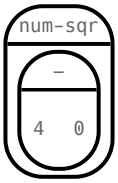
Distance

The Player is at (4, 2) and the Target is at (0, 5).

Distance takes in the player's x, player's y, character's x and character's y. Use the formula below to fill in the EXAMPLE:

$$\sqrt{(4-0)^2+(2-5)^2}$$

Convert it into a Circle of Evaluation. (We've already gotten you started!)



Convert it to Pyret code.

Word Problem: distance

Directions: Write a function `distance`, which takes FOUR inputs: (1) px: The x-coordinate of the player, (2) py: The y-coordinate of the player, (3) cx: The x-coordinate of another game character, (4) cy: The y-coordinate of another game character. It should return the distance between the two, using the Distance formula: $\text{Distance}^2 = (px - cx)^2 + (py - cy)^2$

Contract and Purpose Statement

Every contract has three parts…

`#` _____ `::` _____ `->` _____

function name — *Domain* — *Range*

`#` _____

what does the function do?

Examples

Write some examples, then circle and label what changes…

`examples:`

_____ `(` _____ `)` **`is`** _____

function name — *input(s)* — *what the function produces*

_____ `(` _____ `)` **`is`** _____

function name — *input(s)* — *what the function produces*

`end`

Definition

Write the definition, giving variable names to all your input values…

`fun` _____ `(` _____ `):`

function name — *variable(s)*

_____

what the function does with those variable(s)

`end`

Word Problem: is-collision

Directions: Write a function `is-collision`, which takes FOUR inputs: (1) px: The x-coordinate of the player, (2) py: The y-coordinate of the player, (3) cx: The x-coordinate of another game character, (4) cy: The y-coordinate of another game character. It should return true if the coordinates of the player are within **50 pixels** of the coordinates of the other character. Otherwise, false.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

_____ (_____) **is** _____
function name *input(s)* *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values…

fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Notes

Making Pong

Nested Structures

Timers

Directions:

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

`examples:`

_____ (_____) **`is`** _____
function name *input(s)* *what the function produces*

_____ (_____) **`is`** _____
function name *input(s)* *what the function produces*

`end`

Definition

Write the definition, giving variable names to all your input values…

`fun` _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

`end`

Directions:

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ -> _____
function name *Domain* *Range*

_____
what does the function do?

Examples

Write some examples, then circle and label what changes…

`examples:`

_____ (_____) **`is`** _____
function name *input(s)* *what the function produces*

_____ (_____) **`is`** _____
function name *input(s)* *what the function produces*

`end`

Definition

Write the definition, giving variable names to all your input values…

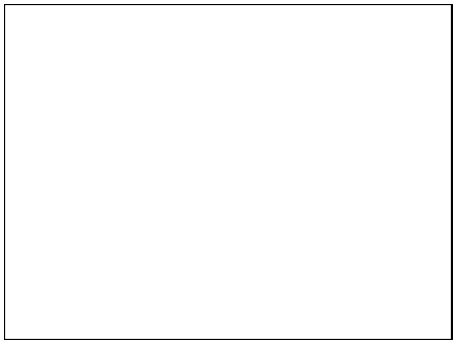
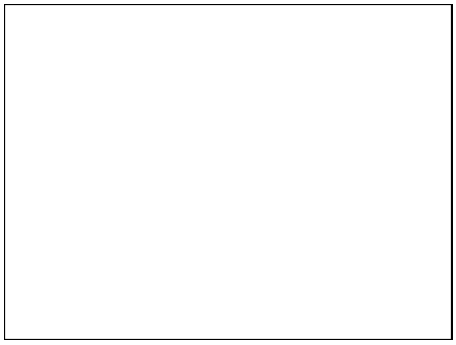
`fun` _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

`end`

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

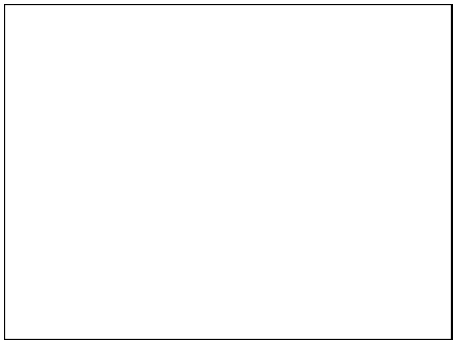
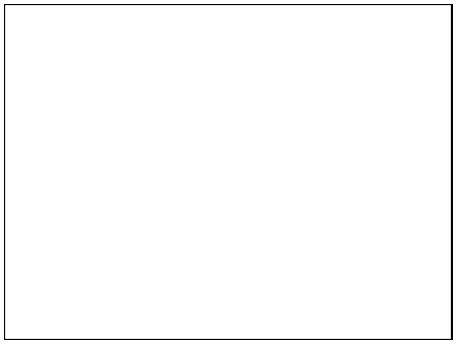
Field name (dangerX, score, playerIMG …)	Datatype (Number, String, Image, Boolean …)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

```
# a _____State is _____________________________________
{}data _____State:
{}| _____ (_____
{}                  _____
{}                  _____
{}                  _____)
{}end

_____ = _____

{}_____ = _____
{}_____ = _____

_____

{}_____
{}_____
{}_____
{}_____
```

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

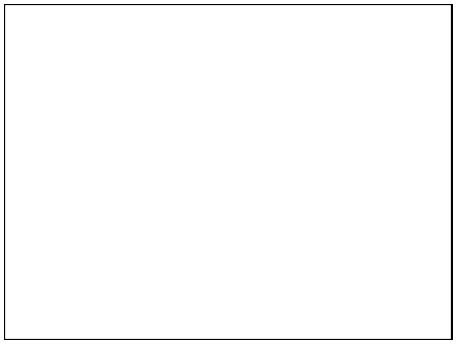
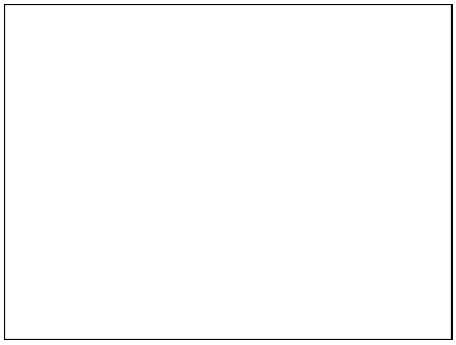
Field name (dangerX, score, playerIMG …)	Datatype (Number, String, Image, Boolean …)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

```
# a _____State is _____________________________________
{}data _____State:
{}| _____ (_____
{}                  _____
{}                  _____
{}                  _____)
{}end

_____ = _____

{}_____ = _____
{}_____ = _____

_____

{}_____
{}_____
{}_____
{}_____
```

Animation Data Worksheet

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

		
Sketch A	**Sketch B**	**Sketch C**

Thing	Describe how it changes

Field name (dangerX, score, playerIMG …)	Datatype (Number, String, Image, Boolean …)

Component	*When is there work to be done?*	To-Do	Done
Data Structure	*If any new field(s) were added, changed, or removed*	☐	☐
draw-state	*If something is displayed in a new way or position*	☑	☐

```
# a _____State is _____________________________________
{}data _____State:
{}| _____ (_____
{}                  _____
{}                  _____
{}                  _____)
{}end

_____ = _____

{}_____ = _____
{}_____ = _____

_____

{}_____
{}_____
{}_____
{}_____
```

Contracts

Contracts tell us how to use a function. For example: `num-sqr :: (n :: Number) -> Number` tells us that the name of the function is `num-sqr`, it takes one input (a `Number`), and it evaluates to a `Number`. From the contract, we know `num-sqr(4)` will evaluate to a `Number`.

Name		Domain		Range
`# triangle`	`::`	`(side-length :: Number, style :: String, color :: String)`	`->`	`Image`
`#`				
`# circle`	`::`	`(radius :: Number, style :: String, color :: String)`	`->`	`Image`
`#`				
`# star`	`::`	`(radius :: Number, style :: String, color :: String)`	`->`	`Image`
`#`				
`# rectangle`	`::`	`(width :: Num, height :: Num, style :: Str, color :: Str)`	`->`	`Image`
`#`				
`# ellipse`	`::`	`(width :: Num, height :: Num, style :: Str, color :: Str)`	`->`	`Image`
`#`				
`# square`	`::`	`(size-length :: Number, style :: String, color :: String)`	`->`	`Image`
`#`				
`# text`	`::`	`(str :: String, size :: Number, color :: String)`	`->`	`Image`
`#`				
`# overlay`	`::`	`(img1 :: Image, img2 :: Image)`	`->`	`Image`
`#`				
`beside`	`::`	`Image, Image`	`->`	`Image`
`beside(star(50, "solid", "orange"),circle(50, "solid", "green"))`				

Contracts

Contracts tell us how to use a function. For example: `num-sqr :: (n :: Number) -> Number` tells us that the name of the function is `num-sqr`, it takes one input (a `Number`), and it evaluates to a `Number`. From the contract, we know `num-sqr(4)` will evaluate to a `Number`.

Name		Domain		Range
`above`	`::`	`Image, Image`	`->`	`Image`
`above(triangle(30, "solid", "red"),square(30, "solid", "blue"))`				
`# above`	`::`	`(img1 :: Image, img2 :: Image)`	`->`	`Image`
`#`				
`# put-image`	`::`	`(img1 :: Image, x :: Number, y :: Number, img2 :: Image)`	`->`	`Image`
`#`				
`# rotate`	`::`	`(degree :: Number, img :: Image)`	`->`	`Image`
`#`				
`# scale`	`::`	`(factor :: Number, img :: Image)`	`->`	`Image`
`#`				
`# string-repeat`	`::`	`(text :: String, repeat :: Number)`	`->`	`String`
`#`				
`# string-contains`	`::`	`(text :: String, search-for :: String)`	`->`	`Boolean`
`#`				
`# num-sqr`	`::`	`(n :: Number)`	`->`	`Number`
`#`				
`# num-sqrt`	`::`	`(n :: Number)`	`->`	`Number`
`#`				

Contracts

Contracts tell us how to use a function. For example: `num-sqr :: (n :: Number) -> Number` tells us that the name of the function is `num-sqr`, it takes one input (a `Number`), and it evaluates to a `Number`. From the contract, we know `num-sqr(4)` will evaluate to a `Number`.

Name		Domain		Range
`# num-min`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`#`				
`# num-max`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`#`				
`# string-equal`	`::`	`(str1 :: String, str2 :: String)`	`->`	`Boolean`
`#`				
`# and`	`::`	`(test1 :: Boolean, test2 :: Boolean)`	`->`	`Boolean`
`#`				
`# or`	`::`	`(test1 :: Boolean, test2 :: Boolean)`	`->`	`Boolean`
`#`				
`#`	`::`		`->`	
`#`				
`#`	`::`		`->`	
`#`				
`#`	`::`		`->`	
`#`				
`#`	`::`		`->`	
`#`				

These materials were developed partly through support of the National Science Foundation, (awards 1042210, 1535276, 1648684, and 1738598), and are licensed under a Creative Commons 4.0 Unported License. Based on a work at www.BootstrapWorld.org. Permissions beyond the scope of this license may be available by contacting contact@BootstrapWorld.org.