Name: _____

Student Workbook

Spring, 2021 - Pyret Edition

Workbook v3.0

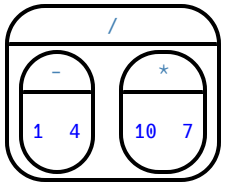
Brought to you by the Bootstrap team:

- Emmanuel Schanzer
- Kathi Fisler
- Shriram Krishnamurthi
- Dorai Sitaram
- Joe Politz
- Jennifer Poole
- Ed Campos
- Ben Lerner
- Flannery Denny

Visual Designer: Colleen Murphy

Order of Operations

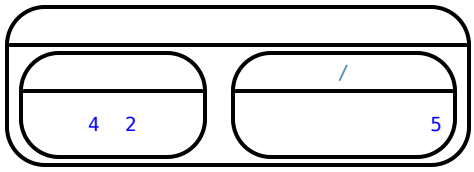
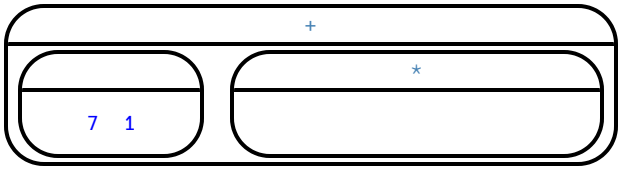
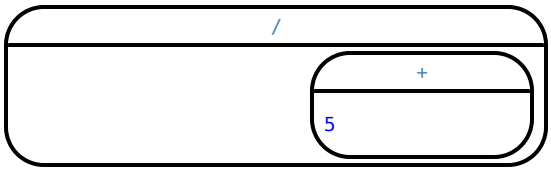
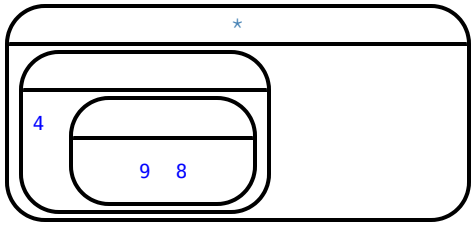
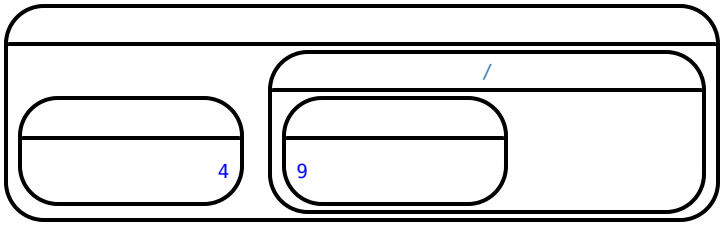
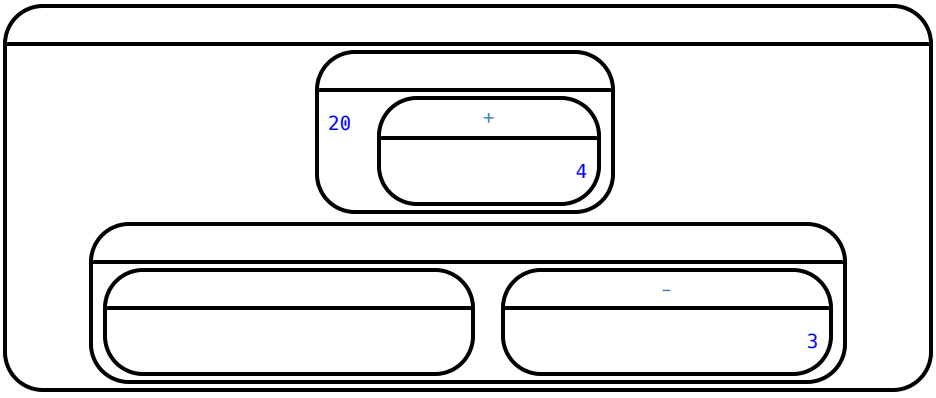
Order of Operations is incredibly important when programming. To help us organize our math into something we can trust, we can *diagram* a math expression using the **Circles of Evaluation**. For example, the expression $1 - 4 \div 10 \times 7$ can be diagrammed as shown below.



To convert a **Circle of Evaluation** into code, we walk through the circle from outside-in, moving left-to-right. We type an open parenthesis when we *start* a circle, and a close parenthesis when we *end* one. Once we're in a circle, we write whatever is on the left of the circle, then the **operation** at the top, and then whatever is on the right. The circle above, for example, would be programmed as `((1 - 4) / (10 * 7))`.

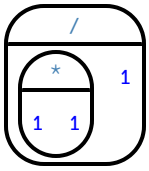
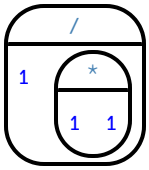
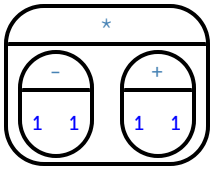
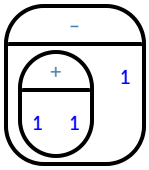
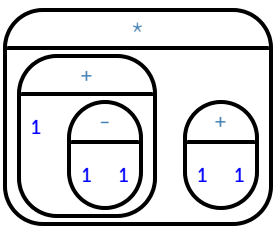
Completing Circles of Evaluation from Arithmetic Expressions

For each expression on the left, finish the Circle of Evaluation on the right by filling in the blanks.

	Arithmetic Expression	Circle of Evaluation
1	$4 + 2 - \frac{10}{5}$	
2	$7 - 1 + 5 \times 8$	
3	$\frac{-15}{5 + -8}$	
4	$(4 + (9 - 8)) \times 5$	
5	$6 \times 4 + \frac{9 - -6}{5}$	
★	$\frac{20}{6 + 4} - \frac{5 \times 9}{-12 - 3}$	

Matching Circles of Evaluation and Arithmetic Expressions

Draw a line from each Circle of Evaluation on the left to the corresponding arithmetic expression on the right.

Circle of Evaluation			Arithmetic Expression
	1	A	$1 \div (1 \times 1)$
	2	B	$(1 + 1) - 1$
	3	C	$(1 \times 1) \div 1$
	4	D	$(1 + (1 - 1)) \times (1 + 1)$
	5	E	$(1 - 1) \times (1 + 1)$

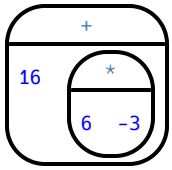
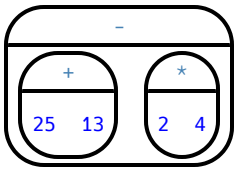
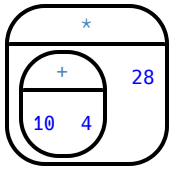
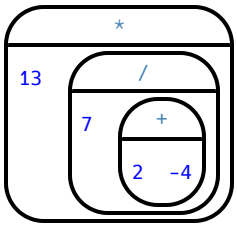
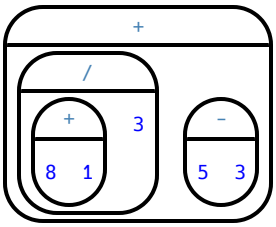
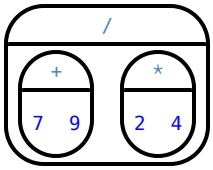
Translate Arithmetic to Circles of Evaluation & Code (Intro)

Translate each of the arithmetic expressions below into Circles of Evaluation, then translate them to Code.

	Arithmetic	Circle of Evaluation	Code
1	$(3 \times 7) - (1 + 2)$		
2	$3 - (1 + 2)$		
3	$3 - (1 + (5 \times 6))$		
4	$(1 + (5 \times 6)) - 3$		

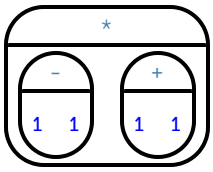
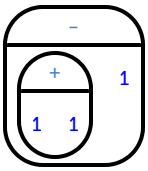
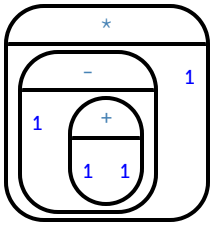
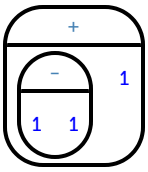
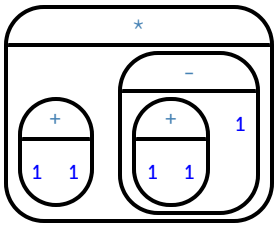
Completing Partial Code from Circles of Evaluation

For each Circle of Evaluation on the left, finish the Code on the right by filling in the blanks.

	Circle of Evaluation	Code
1		(_____ + (6 * _____))
2		((_____ + 13) _____ (_____ _____ 4))
3		((_____ + 4) _____ _____ _____)
4		(13 _____ (7 _____ (2 _____ -4)))
5		(((8 _____ 1) _____ 3) _____ (5 _____ 3))
6		((_____ + _____) / (_____ * _____))

Matching Circles of Evaluation & Code

Draw a line from each Circle of Evaluation on the left to the corresponding Code on the right.

Circle of Evaluation			Code
	1	A	`((1 - (1 + 1)) * 1)`
	2	B	`((1 - 1) * (1 + 1))`
	3	C	`((1 + 1) * ((1 + 1) - 1))`
	4	D	`((1 + 1) - 1)`
	5	E	`((1 - 1) + 1)`

Translate Arithmetic to Circles of Evaluation & Code 2

Translate each of the arithmetic expressions below into Circles of Evaluation, then translate them to Code.

	Arithmetic	Circle of Evaluation	Code
1	$6 \times 8 + (7 - 23)$		
2	$18 \div 2 + 24 \times 4 - 2$		
3	$22 - 7 \div 3 + 2$		
4	$24 \div 4 \times 2 - 6 + 20 \times 2$		

Arithmetic Expressions to Circles of Evaluation & Code - Challenge

Translate each of the arithmetic expressions below into Circles of Evaluation, then translate them to Code.

	Arithmetic	Circle of Evaluation	Code
1	$\frac{16 + 3^2}{\sqrt{49} - 2}$		
2	$45 - 9 \times (3 + (2 - 4)) - 7$		
3	$50 \div 5 \times 2 - ((3 + 4) \times 2 - 5)$		

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) editor, 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.

Strings

String values are always in quotes.

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

---

6) Try typing your name *(in quotes!)*.

7) Try typing a sentence like "I'm excited to learn to code!" *(in quotes!)*.

8) Try typing your name with the opening quote, but *without the closing quote.* Read the error message!

9) Now try typing your name *without any quotes.* Read the error message!

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

Operators

11) 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?

---

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

---

13) 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:	Computer Returns:		Prediction:	Computer Returns:
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:	Computer Returns:
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 **Images** are there in the entire universe?

21) 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? (Numbers? Strings? Booleans?)	

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.

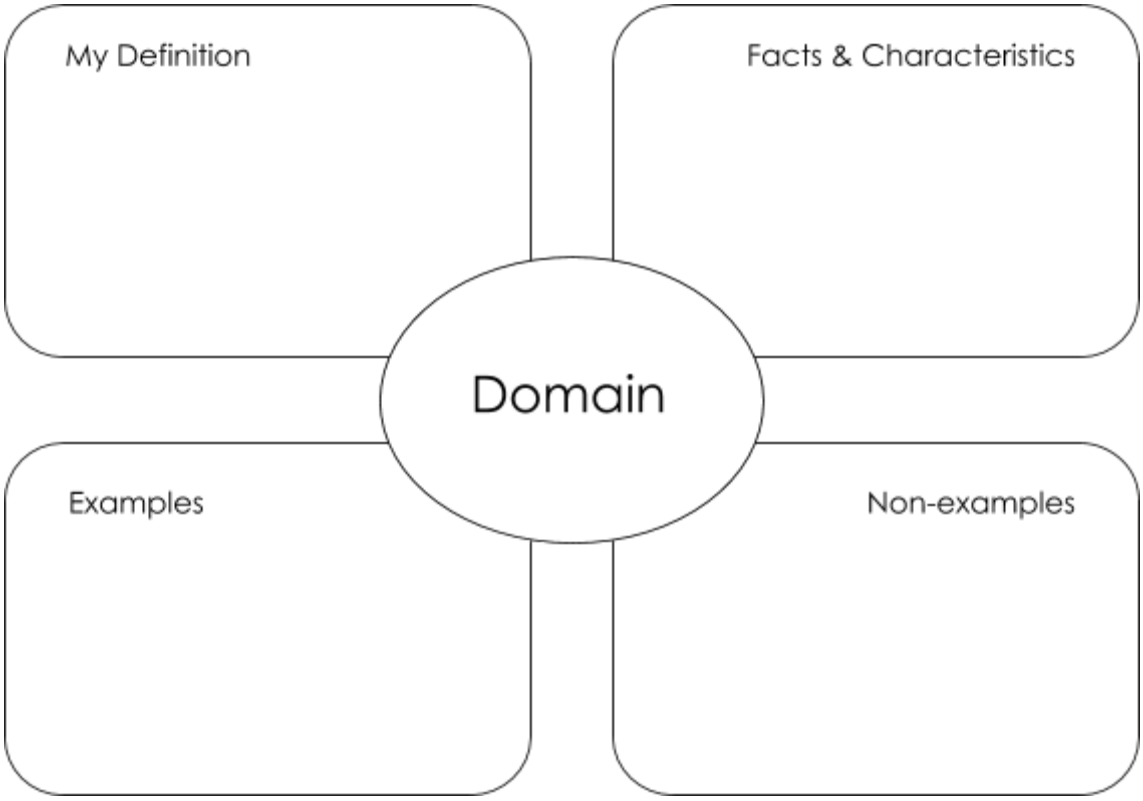
Can you spot the mistake? _____

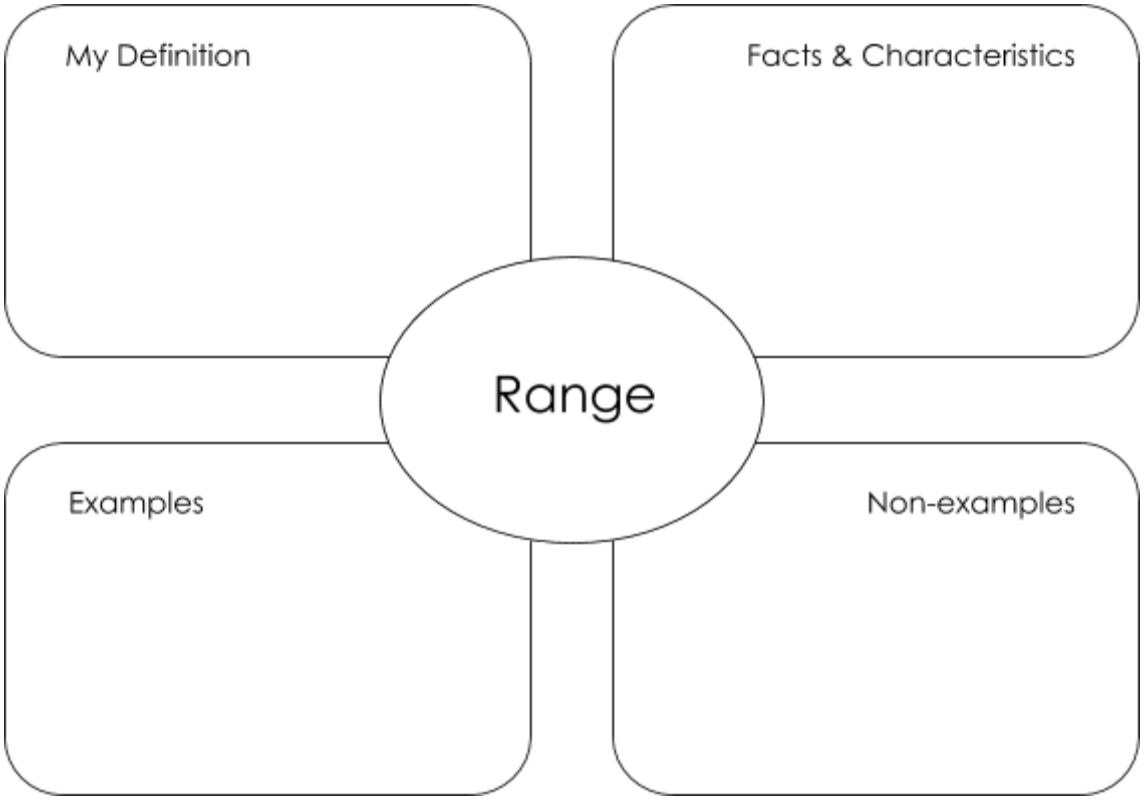
8) `triangle (20, "solid", "red")`

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 may 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")`

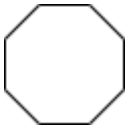
Using Contracts

Use the contracts to write expressions to generate images similar to those pictured.

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

	
	
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.

_____

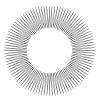
_____

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.

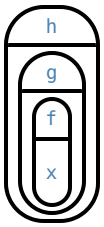
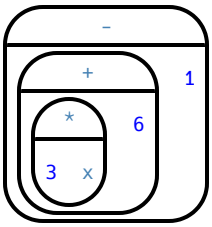
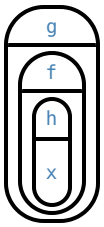
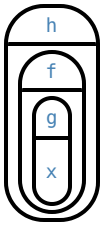
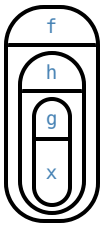
Image			Expression
	1	A	`radial-star(5, 50, 200, "solid", "black")`
	2	B	`radial-star(7, 100, 200, "solid", "black")`
	3	C	`radial-star(7, 100, 200, "outline", "black")`
	4	D	`radial-star(10, 150, 200, "solid", "black")`
	5	E	`radial-star(10, 20, 200, "solid", "black")`
	6	F	`radial-star(100, 20, 200,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

Circle of Evaluation:

Code: _____

3) A solid, green star, that is half the size of the original (using `scale`)

Circle of Evaluation:

Code: _____

4) A solid, green star of size 50 that has been rotated 45 degrees counter-clockwise

Circle of Evaluation:

Code: _____

5) A solid, green star that is 3 times the size of the original **and** has been rotated 45 degrees

Circle of Evaluation:

Code: _____

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".
Circle of Evaluation:

Code: _____

3) The "image of your name" flipped vertically.
Circle of Evaluation:

Code: _____

4) The "image of your name" above "the image of your name" flipped vertically.
Circle of Evaluation:

Code: _____

5) The "image of your name" flipped horizontally beside "the image of your name".
Circle of Evaluation:

Code: _____

Function Composition — scale-xy

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

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

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

The Image:	Circle of Evaluation:	Code:
	rhombus 40 90 "solid" "purple"	`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.

Circle of Evaluation:

Code: _____

2) A purple rhombus that is stretched 4 times as tall

Circle of Evaluation:

Code: _____

3) The tall rhombus overlayed on the wide rhombus.

Circle of Evaluation:

Code: _____

★: Overlay a red rhombus onto the last image you made.

Circle of Evaluation:

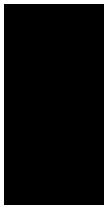
Code: _____

More than one way to Compose an Image!

Read through these 4 expressions and try to picture the images they are composing. If you're not sure what they'll look like, type them into the interactions area of your editor and see if you can figure out how the code connects to the 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		• `rotate(90, rectangle(200, 100, "solid", "black"))` • _____ • _____
2		• `above(rectangle(200, 100, "solid", "black"), rectangle(200, 100, "solid", "black"))` • _____ • _____
3		• `scale(0.5, rectangle(600, 200, "solid", "black"))` • _____ • _____
★		• `overlay(rectangle(100, 200, "solid", "black"), rectangle(200, 100, "solid", "black"))` • _____ • _____

Defining Values

In math, we use **values** like $-98.1, 2/3$ amd 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 *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?

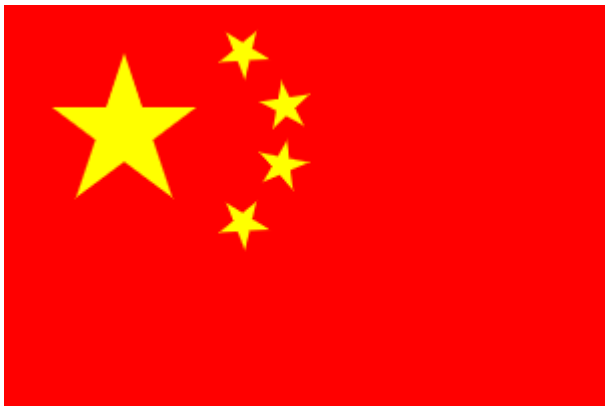
Look at the expressions listed below. Think about what you expect each of them to produce. Then, test them out one at a time in the Interactions Area.

- `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) Highlight or circle all instances of the structure that makes the repeated image in the code below.

3) In the code below, highlight or circle all instances of the expression for that image.

```
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"))))))
```

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

---

5) Open the Chinese flag starter file (Pyret) and click Run.

Then type `china` into the interactions area and click **Enter**.

6) **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.

7) Now change the color of all of the stars to black, in both files. Then change the size of the stars.

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

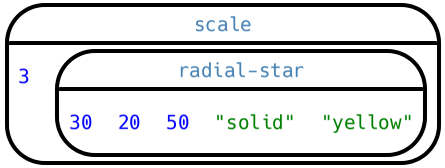
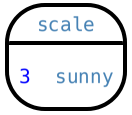
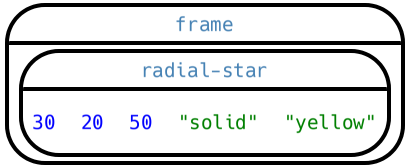
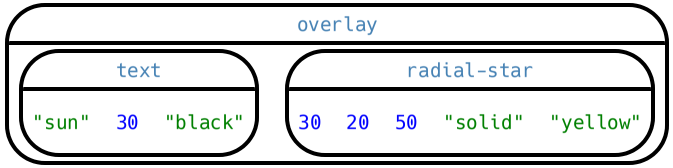
Challenge:

- 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.

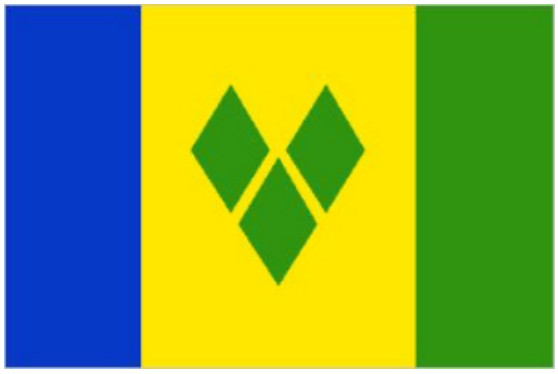
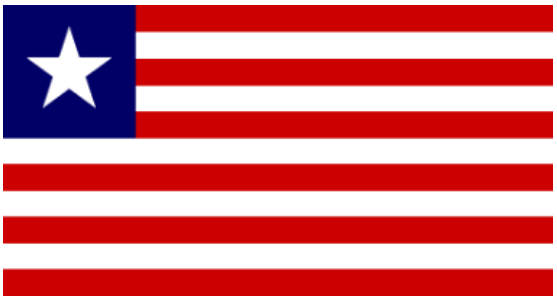
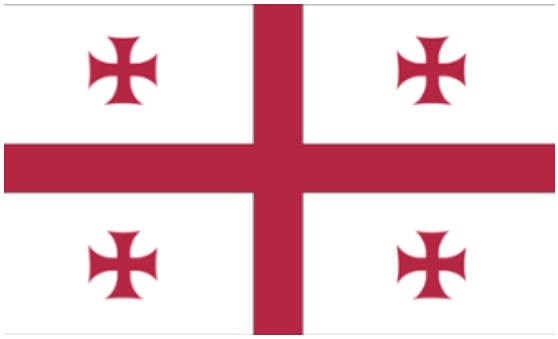
2) Write the code to define the value of `sunny`. _____

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

Think of the background image as a sheet of graph paper with the origin (0,0) in the bottom left corner.
The numbers in `put-image` specify a point on that graph paper, where the center of the top image should be placed.

The width of the rectangle is 300 and the height is 200. The definitions for `dot` and `background` are:

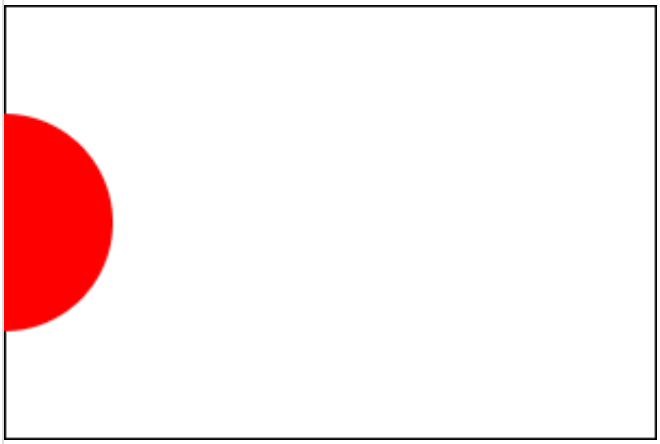
```
dot = circle(50, "solid", "red")
background = rectangle(300, 200, "outline", "black")
```

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

A

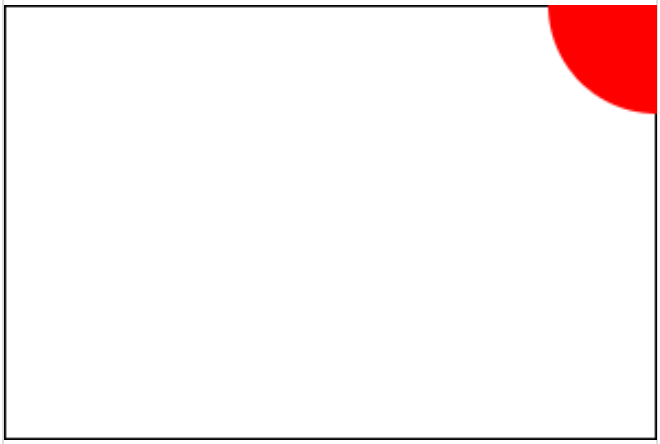
put-image(dot, _____, _____ background)

B



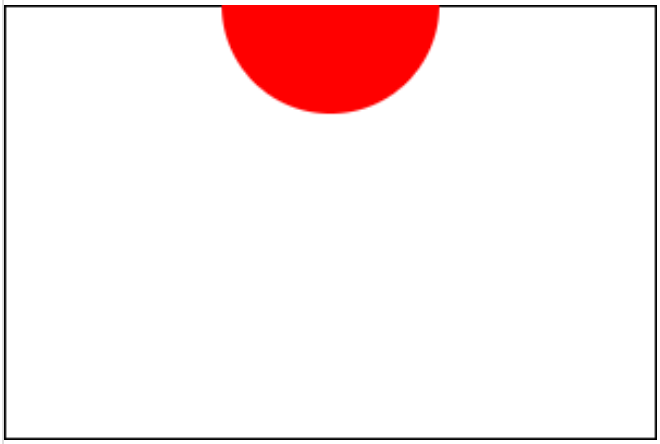
put-image(dot, _____, _____ background)

C



put-image(dot, _____, _____ background)

D



put-image(dot, _____, _____ background)

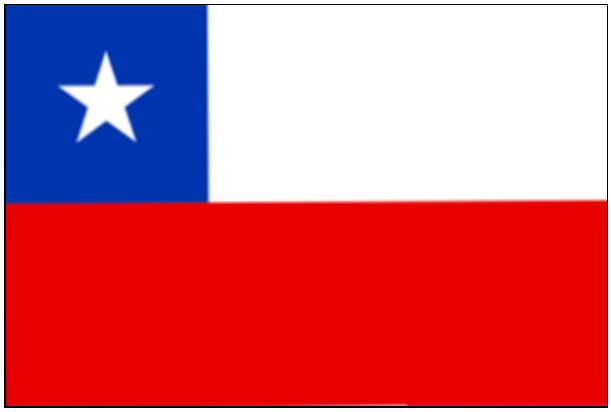
Decomposing Flags

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)

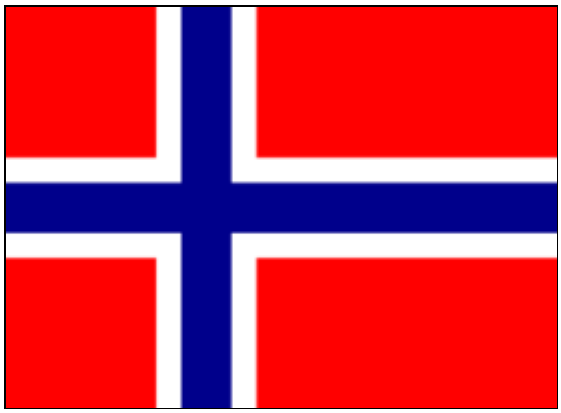


shape:	color:	width:	height:	x	y

Panama (300 x 200)

shape:	color:	width:	height:	x	y

Norway (330 x 240)



shape:	color:	width:	height:	x	y

Defining Functions

Functions can be viewed in *multiple representations*. You already know one of them: ***Contracts***, which specify the Name, Domain, and Range of a function. Contracts are a way of thinking of functions as a *mapping* between one set of data and another. For example, a mapping from Numbers to Strings:

```
f :: Number -> String
```

Another way to view functions is with ***Examples***. Examples are essentially input-output tables, showing what the function would do for a specific input:

In our programming langauge, we focus on the last two columns and write them as code:

```
examples:
  f(1) is 1 + 2
  f(2) is 2 + 2
  f(3) is 3 + 2
  f(4) is 4 + 2
end
```

Finally, we write a formal **function definition** ourselves. The pattern in the Examples becomes *abstract* (or "general"), replacing the inputs with ***variables***. In the example below, the same definition is written in both math and code:

$$f(x) = x + 2$$

```
fun f(x): x + 2 end
```

Look for connections between these three representations!

- The function name is always the same, whether looking at the Contract, Examples, or Definition.
- The number of inputs in the Examples is always the same as the number of types in the Domain, which is always the same as the number of variables in the Definition.
- The "what the function does" pattern in the Examples is almost the same in the Definition, but with specific inputs replaced by variables.

Matching Examples and Definitions (Math)

Look at each set of examples on the left and circle what is changing from one example to the next.
Then, *match* the examples on the left to the definitions on the right.

Examples:		Functions:	

x	$f(x)$
1	2×1
2	2×2
3	2×3

1

A $f(x) = x - 3$

x	$f(x)$
15	$15 - 3$
25	$25 - 3$
35	$35 - 3$

2

B $f(x) = 2x$

x	$f(x)$
10	$10 + 2$
15	$15 + 2$
20	$20 + 2$

3

C $f(x) = 2x + 1$

x	$f(x)$
0	$3(0) - 2$
1	$3(1) - 2$
2	$3(2) - 2$

4

D $f(x) = 3x - 2$

x	$f(x)$
10	$2(10) + 1$
20	$2(20) + 1$
30	$2(30) + 1$

5

E $f(x) = x + 2$

Matching Examples and Function Definitions

Highlight the variables in `gt` and label them with the word "size".

```
examples:
  gt(20) is
    triangle(20, "solid", "green")
  gt(45) is
    triangle(45, "solid", "green")
end
```

```
fun gt(size): triangle(size, "solid", "green") end
```

Highlight and label the variables in the example lists below. Then, using `gt` as a model, match the examples to their corresponding function definitions.

Examples		Definition

`examples:`
`  f("solid") is`
`  circle(8, "solid", "red")`
`  f("outline") is`
`  circle(8, "outline", "red")`
`end`
	1	A `fun f(s): star(s, "outline", "red") end`

`examples:`
`  f(2) is 2 + 2`
`  f(4) is 4 + 4`
`  f(5) is 5 + 5`
`end`
	2	B `fun f(num): num + num end`

`examples:`
`  f("red") is circle(7, "solid", "red")`
`  f("teal") is`
`  circle(7, "solid", "teal")`
`end`
	3	C `fun f(c): star(9, "solid", c) end`

`examples:`
`  f("red") is star(9, "solid", "red")`
`  f("grey") is star(9, "solid", "grey")`
`  f("pink") is star(9, "solid", "pink")`
`end`
	4	D `fun f(s): circle(8, s, "red") end`

`examples:`
`  f(3) is star(3, "outline", "red")`
`  f(8) is star(8, "outline", "red")`
`end`
	5	E `fun f(c): circle(7, "solid", c) end`

Matching Examples and Contracts

Match each set of examples (left) with the contract that best describes it(right).

Examples			Contract

`examples:`
`  f(5) is 5 / 2`
`  f(9) is 9 / 2`
`  f(24) is 24 / 2`
`end`
	1	A	`# f :: Number -> Number`

`examples:`
`  f(1) is`
`  rectangle(1, 1, "outline", "red")`
`  f(6) is`
`  rectangle(6, 6, "outline", "red")`
`end`
	2	B	`# f :: String -> Image`

`examples:`
`  f("pink", 5) is`
`  star(5, "solid", "pink")`
`  f("blue", 8) is`
`  star(8, "solid", "blue")`
`end`
	3	C	`# f :: Number -> Image`

`examples:`
`  f("Hi!") is text("Hi!", 50, "red")`
`  f("Ciao!") is text("Ciao!", 50, "red")`
`end`
	4	D	`# f :: Number, String -> Image`

`examples:`
`  f(5, "outline") is`
`  star(5, "outline", "yellow")`
`  f(5, "solid") is`
`  star(5, "solid", "yellow")`
`end`
	5	E	`# f :: String, Number -> Image`

Contracts, Examples & Definitions

gt

Directions: Define a function called `gt`, which makes solid green triangles of whatever size we want.

Every contract has three parts…

```
# gt :: Number -> Image
```
function name *domain* *range*

Write some examples, then circle and label what changes…

examples:

```
gt ( 10 ) is triangle(10, "solid", "green")
```
function name *input(s)* *what the function produces*

```
gt ( 20 ) is triangle(20, "solid", "green")
```
function name *input(s)* *what the function produces*

end

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

```
fun gt ( size ):
  triangle(size, "solid", "green")
```
function name *variable(s)*

what the function does with those variable(s)

end

bc

Directions: Define a function called `bc`, which makes solid blue circles of whatever radius we want.

Every contract has three parts…

```
# _____ :: _____ -> _____
```
function name *domain* *range*

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

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

What's on your mind?

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!

Creating Contracts From Examples

Write the contracts used to create each of the following collections of examples.

1)

```
examples:
  big-triangle(100, "red") is
    triangle(100, "solid", "red")
  big-triangle(200, "orange") is
    triangle(200, "solid", "orange")
end
```

2)

```
examples:
  purple-square(15) is
    rectangle(15, 15, "outline", "purple")
  purple-square(6) is
    rectangle(6, 6, "outline", "purple")
end
```

3)

```
examples:
  banner("Game Today!") is
    text("Game Today!", 50, "red")
  banner("Go Team!") is
    text("Go Team!", 50, "red")
  banner("Exit") is
    text("Exit", 50, "red")
end
```

4)

```
examples:
  twinkle("outline", "red") is
    star(5, "outline", "red")
  twinkle("solid", "pink") is
    star(5, "solid", "pink")
  twinkle("outline", "grey") is
    star(5, "outline", "grey")
end
```

5)

```
examples:
  half(5) is 5 / 2
  half(8) is 8 / 2
  half(900) is 900 / 2
end
```

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…

```
# upside-down:: Image -> Image
```

function name | *domain* | *range*

```
# Consumes an image, and flips 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
```

Contract and Purpose Statement

Every contract has three parts…

```
# product-squared:: Number, Number -> Number
```

function name | *domain* | *range*

```
# Consumes two numbers and squares their product
```

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
```

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

Writing Quality Purpose Statements

3 Reads

1st Read: What is this problem about?	2nd Read: What are the Quantities?
3rd Read: What is a good Purpose Statement?	

Stronger & Clearer

Purpose Statement 1st Revision:
Purpose Statement 2nd Revision:

The Design Recipe - Direct Variation

Directions : Write a function `wage` , that takes in a number of hours worked and returns the amount a worker will get paid if their rate is $10.25/hr.

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 : On average, people burn about 11 calories/minute riding a bike. Write a function `calories-burned` that takes in the number of minutes you bike and returns the number of calories burned. .

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

The Design Recipe (Practice 1)

Directions: Write a function `marquee` that takes in a message and returns that message in large gold letters.

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: Write a function `num-cube` that takes in a number and returns the cube of that number.

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

The Design Recipe (Practice 2)

Directions: Write a function `split-tab` that takes in a cost and the number of people sharing the bill and splits the cost equally.

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: Write a function `tip-calculator` that takes in the cost of a meal and returns the 15% tip for that meal.

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

The Design Recipe (Practice 3)

Directions: The Swamp in the City Festival is ordering t-shirts. The production cost is $75 to set up the silk screen and $9 per shirt. Write a function `min-shirt-price` that takes in the number of shirts to be ordered, *n*, and returns the minimum amount the festival should charge for the shirts in order to break even. (Assume that they will sell all of the shirts.)

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

The Design Recipe (Slope/Intercept 1)

Directions: For his birthday, James' family decided to open a savings account for him. He started with $50 and committed to adding $10 a week from his afterschool job teaching basketball to kindergartners. Write a function `savings` that takes in the number of weeks since his birthday and calculates how much money he has saved.

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: Write a function `moving` that takes in the days and number of miles driven and returns the cost of renting a truck. The truck is $45 per day and each driven mile is 15¢.

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

The Design Recipe (Negative Slope/Intercept)

Directions : An Olympic pool holds 660,000 gallons of water. A fire hose can spray about 250 gallons per minute. Write a function `pool` that takes in the number of minutes that have passed and calculates how much water is still needed to fill it.

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 : The community arts fund awards a $1500 grant each month to support a new mural. They started with $50000 in their account. Write a function `funds-available` that takes in the number of months and calculates how much money they have left.

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

The Design Recipe (Geometry - Rectangles)

Directions: Write a function `lawn-area` that takes in the length and width of a rectangular lawn and returns its area.

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: Write a function `rect-perimeter` that takes in the length and width of a rectangle and returns the perimeter of that rectangle.

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

The Design Recipe (Geometry - Rectangular Prisms)

Directions: Write a function `rectprism-vol` that takes in the length, width, and height of a rectangular prism and returns the Volume of a rectangular prism.

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: Write a function `rect-prism-sa` that takes in the width, length and height of a rectangular prism and calculates its surface area (the sum of the areas of each of its six faces)

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

The Design Recipe (Geometry - Circles)

Directions: Write a function `circle-area-dec` that takes in a radius and uses the decimal approximation of pi (3.14) to return the area of the circle.

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: Write a function `circumference` that takes in a radius and uses the decimal approximation of pi (3.14) to return the circumference of the circle.

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

The Design Recipe (Geometry - Cylinders)

Directions: Write a function `circle-area` that takes in a radius and uses the fraction approximation of pi ($\frac{22}{7}$) to return the area of the circle.

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: Write a function `cylinder` that takes in a cylinder's radius and height and calculates its volume, making use of the function *circle-area* .

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…

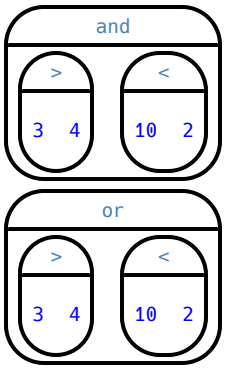
fun _____ (_____):
function name *variable(s)*

_____
what the function does with those variable(s)

end

Inequalities

- Sometimes we want to *ask questions* about data. For example, is `x` greater than `y`? Is one string equal to another? These questions can't be answered with a Numbers. Instead, they are answered with a new data type called a **Boolean**.
- Video games use Booleans for many things: asking when a player's health is equal to zero, whether two characters are close enough to bump into one another, or if a character's coordinates put it off the edge of the screen.
- A Boolean value is either `true` or `false`. Unlike Numbers, Strings, and Images, Booleans have only two possible values.
- You already know some functions that produce Booleans, such as `<` and `>`! Our programming language has them, too: `3 < 4`, `10 > 2`, and `-10 == 19`.
- We also have ways of writing **Compound Inequalities**, so we can ask more complicated questions using the **`and`** and **`or`** functions.
  - `(3 > 4) and (10 < 2)` translates to "three is greater than four *and* ten is less than two". This will evaluate to `false`, since the **`and`** function requires that both sub-expressions be `true`.
  - `(3 > 4) or (10 < 2)`, which translates to "three is greater than four *or* ten is less than two". This will evaluate to `true`, since the **`or`** function only requires that one sub-expression be `true`.
- The Circles of Evaluation work the same way with Booleans that they do with Numbers, Strings and Images:



Boolean Functions

Explore the functions in the *Booleans Starter File*. What characteristics define them as Booleans?

Fill in the blanks below so that each of the five functions returns `true`

1) `is-odd (          )`

2) `is-even (          )`

3) `is-less-than-one (          )`

4) `is-continent (          )`

5) `is-primary-color (          )`

Fill in the blanks below so that each of the five functions returns `false`

6) `is-odd (          )`

7) `is-even (          )`

8) `is-less-than-one (          )`

9) `is-continent (          )`

10) `is-primary-color (          )`

Simple Inequalities

Each inequality expression in the first column contains a number.
Decide whether or not that number is a solution to the expression and place it in the appropriate column.
Then identify 4 *solution* and 4 *non-solution* values for `x`.

- **Solutions** will make the expression `true`.
- **Non-Solutions** will make the expression `false`.

Challenge yourself to use negatives, positives, fractions, decimals, etc. for your `x` values.

Expression	4 solutions that evaluate to `true`	4 non-solutions that evaluate to `false`
`x > 2`		
`x <= -2`		
`x < 3.5`		
`x >= -1`		
`x > -4`		
`x <> 2`		

1) For which inequalities was the number from the expression part of the solution?

_____

2) For which inequalities was the number from the expression not part of the solution?

_____

3) For which inequalities were the solutions on the left end of the number line?

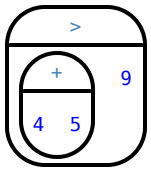
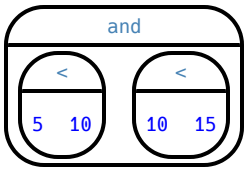
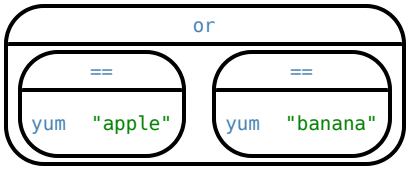
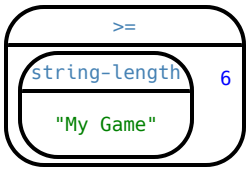
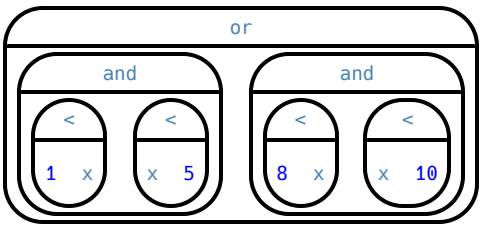
_____

4) For which inequalities were the solutions on the right end of the number line?

_____

Converting Circles of Evaluation to Code

For each Circle of Evaluation on the left-hand side, write the code for the Circle on the right-hand side

	Circle of Evaluation	Code
1		
2		
3		
4		
5		

Compound Inequalities — Practice

Create the Circles of Evaluation, then convert the expressions into code in the space provided.

1) 2 is less than 5, and 0 is equal to 6

What will this evaluate to? _____

2) 6 is greater than 8, or -4 is less than 1

What will this evaluate to? _____

3) The String “purple” is the same as the String “blue”, and 3 plus 5 equals 8

What will this evaluate to? _____

4) Write the contracts for `and` & `or` in your Contracts page.

Compound Inequalities: Solutions & Non-Solutions

For each Compound Inequality listed below, identify 4 *solutions* and 4 *non-solutions*. *If there are* ***no solutions*** *or the solution set includes* ***all real numbers*** *you can write that instead of making a list.*

- Solutions for ***intersections***, which use **and** will make both of the expressions `true`.
- Solutions for ***unions***, which use **or** will make at least one of the expressions `true`.

Pay special attention to the numbers in the sample expression! Challenge yourself to use negatives, positives, fractions, decimals, etc. for your `x` values.

The first two have been done for you - Answers will vary!

Expression	4 solutions that evaluate to `true`	4 non-solutions that evaluate to `false`
`x > 5` and `x < 15`	6, 9.5, 12, 14.9	-2, 5, 15, 16.1
`x > 5` or `x < 15`	All real numbers	No non-solutions
`x <= -2` and `x > 7`		
`x <= -2` or `x > 7`		
`x < 3.5` and `x > -4`		
`x < 3.5` or `x > -4`		
`x >= -1` and `x > -5`		
`x >= -1` or `x > -5`		
`x < -4` and `x > 2`		

1) Could there ever be a union with *no solutions*? Explain your thinking.

2) Could there ever be an intersection whose solution is *all real numbers*? Explain your thinking.

Compound Inequality Functions

Each of the plots below was generated using the code `inequality(comp-ineq, [list: -1, 0, 1.6, 3, 5.2, 7, 8.1, 9] )`. With the exception of the example, each plot below was defined using the numbers 3 and 7. Write the code for how `comp-ineq` was defined for each plot in the space provided.

code: `fun comp-ineq(x): (x > 0) and (x <= 8.1) end`

code: _____

code: _____

code: _____

code: _____

Sam the Butterfly

Open the "Sam the Butterfly" starter file and press "Run". *(Hi, Sam!)*
Move Sam around the screen using the arrow keys.

1) What do you notice about the program?

2) What do you wonder?

3) What do you see when Sam is at (0,0)? Why is that?

4) What changes as the butterfly moves left and right?

Sam is in a 640 × 480 yard. Sam's mom wants Sam to stay in sight.

How far to the left and right can Sam go and still remain visible?

Use the new inequality functions to answer the following questions *with code* :

5) Sam hasn't gone off the left edge of the screen as long as… _____

6) Sam hasn't gone off the right edge of the screen as long as… _____

7) Use the space below to draw Circles of Evaluation for these two expressions:

Left and Right

Directions: Use the Design Recipe to write a function `is-safe-left`, which takes in an x-coordinate and checks to see if it is greater than -50.

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: Use the Design Recipe to write a function `is-safe-right`, which takes in an x-coordinate and checks to see if it is less than 690.

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-onscreen

Directions: Use the Design Recipe to write a function `is-onscreen`, which takes in an x-coordinate and checks to see if Sam is safe on the left while also being safe on the right.

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

Problem Decomposition

- Sometimes a problem is too complicated to solve all at once. Maybe there are too many variables, or there is just so much information that we can't get a handle on it!
- We can use **Problem Decomposition** to break those problems down into simpler pieces, and then work with the pieces to solve the whole. There are two strategies we can use for decomposition:
  - **Top-Down** - Start with the "big picture", writing functions or equations that describe the connections between parts of the problem. Then, work on defining those parts.
  - **Bottom-Up** - Start with the smaller parts, writing functions or equations that describe the parts we understand. Then, connect those parts together to solve the whole problem.
- You may find that one strategy works better for some types of problems than another, so make sure you're comfortable using either one!

The Design Recipe: Revenue & Cost

Directions: Use the Design Recipe to write a function `revenue`, which takes in the number of glasses sold at $1.75 apiece and calculates the total revenue.

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: Use the Design Recipe to write a function `cost`, which takes in the number of glasses sold and calculates the total cost of materials if each glass costs $.30 to make.

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: profit

Directions: Use the Design Recipe to write a function `profit` that calculates total profit from glasses sold, which is computed by subtracting the total cost from the total revenue.

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

Permutation and Combination

- What are the odds of guessing someone's 8-digit password?
- How many bouquets can we make choosing 4 different flowers from a collection of 10?
- If 10 runners enter a road race, how many different ways can they be ranked?
- If you pick two cards from a deck and they're both queens, what are the odds that the next card will be a queen?

Each of these questions deals with ***permutation*** or ***combination***. Both concepts play a big role in probability and statistics. If you know how many possible outcomes there *could* be, you can predict what your chances are. This is useful for competitive gaming, conducting surveys, and cybersecurity!

Permutation involves computing the number of different ways the same set of things can be re-arranged. If you have a dozen different doughnuts to choose from, how many different ways are there of *arranging* six of them?

Combination involves computing the number of different *subsets* you can make from the same set of things. If you have a dozen doughnuts to choose from, how many different half-dozen choices could you make?

Tree Diagrams

1) The Lopez family loves to go camping. So, each year Grandma buys Savannah and Rosa new gear. This year they're getting camping mugs. They are available in 5 colors: green, blue, red, silver and copper. Draw the tree-diagram for all possible mug permutations, if Grandma chooses a mug for one girl and then the other.

Permutation With Replacement

2) How many different permutations (with replacement) are there for these mugs?

3) Grandma wonders if maybe the mugs should be different colors so that the girls can tell them apart. Draw the tree-diagram for all possible mug permutations, if Grandma chooses a mug for one girl and then the other.

Permutation Without Replacement

4) How many different permutations (without replacement) are there for these mugs?

Permutation

For each of the problems below, (1) figure out whether this involves permutation with or without replacement, then (2) compute the solution. The first one has been done for you.

	Word problem	Replacement?	Solution
1	Joy has picked out seven outfits for the week. She intends to wear each of them once, but she hasn't chosen an order yet. How many different ways could she dress up this week?	Yes ***No***	$$permute\text{-}no\text{-}replace(7,7) = \frac{7!}{(7-7)!}$$ $$= \frac{7!}{(0)!} = \frac{5040}{1} = 5040$$
2	Mrs. Burke's cell phone has a 6-character password. Her son is trying to unlock it to play a game. How many possible passwords does he have to guess?	Yes No	
3	The dentist has 8 different stickers to give away to the next patients `A` through `H`. How many different ways could she give them out?	Yes No	
4	Eric Allatta is the head chef at the top restaurant in Santa Fe. His speciality is four-color enchilada platter, with each enchilada covered in a different sauce. How many ways can he order them on the plate?	Yes No	
5	A magician opens a fresh deck of 52 cards, and asks an audience member to pick six of them. He says he'll guess all six - in order. What are the chances he'll guess them correctly?	Yes No	
6	Emma is knitting a hat, and each row of stitching can be a different color. She has three different colors of yarn to choose from, and the hat has 30 rows. How many different designs could she make?	Yes No	

Combinations

For each of the problems below, (1) figure out whether this involves combination with or without replacement, then (2) compute the solution.

	Word problem	Replacement?	Solution
1	The shaved ice truck has added six new flavors, and three friends want to sample them. They agree to order different flavors, and then all three will try each flavor. How many possible combinations are there?	Yes ***No***	$combinations(6,3) = \frac{6!}{(6-3)!} \div 3$ $= \frac{6!}{3!} \div 3 = \frac{720}{6} \div 3$ $= 120 \div 3 = 40$
2	A soccer team has 20 players, but only 11 are allowed on the field at once. How many different groups of players can be on the field at one time?	Yes No	
3	A set of pool balls is numbered 1-15. How many different ways are there to choose six balls?	Yes No	
4	Six friends get together to play video games. All the games can only have two players, so they decided to pair off to make sure everyone gets to play everyone else. How many games will they have to play?	Yes No	
5	A set of pool balls is numbered 1-15. Seven of them are striped and eight are solid colors. How many different ways are there to choose 4 balls where 2 are striped and 2 are solid?	Yes No	
★	A pizzeria has a 3-topping special on any pizza, for only $12.99. If they have 10 toppings to choose from, how many different pizzas can they make?	Yes No	

Combination or Permutation?

Look at the word problems below. **Without solving them** , circle whether they are asking for a permutation or a combination?

1	How many ways can the letters in "Kathi" be re-arranged?	Permutation Combination
2	Shriram's favorite football team is lining up to run onto the field. How many different ways can they be ordered?	Permutation Combination
3	Flannery is planning to perform 8 songs at a Cajun music festival, and there are 30 different songs she could play. How many different set lists could she put together?	Permutation Combination
4	How many possible 3-color blends can be made from the seven colors of the rainbow?	Permutation Combination
5	How many 8-letter passwords are there, if no character can be used twice?	Permutation Combination
6	How many different ways are there to set a combination lock?	Permutation Combination
7	If Servane is holding a dozen different cupcakes and wants to give two to her friend, what are the chances that she chooses red velvet and chocolate froster?	Permutation Combination
8	Joy is arranging flowers for a bouqet. The store has 18 different kinds of flowers for her to choose from. If the bouquets each need 10 flowers, how many different bouqets could she make?	Permutation Combination
10	Matthias is making a candy coated in different colors, so that biting into it will "expose the rainbow" (the catchphrase he's chosen). His machine can make any of 8 different colors, but each candy can only be coated four times. How many unique color combinations can you find in these candies?	Permutation Combination

Introduction to Computational Data Science

Many important questions ("What's the best restaurant in town?", "Is this law good for citizens?", etc.) are answered with *data*. Data Scientists try and answer these questions by writing *programs that ask questions about data*.

Data of all types can be organized into **Tables**.

- Every Table has a **header row** and some number of **data rows**.
- **Quantitative data** is numeric and measures *an amount*, such as a person's height, a score on a test, distance, etc. A list of quantitative data can be ordered from smallest to largest.
- **Categorical data** is data that specifies *qualities*, such as sex, eye color, country of origin, etc. Categorical data is not subject to the laws of arithmetic — for example, we cannot take the "average" of a list of colors.

The Animals Dataset

name	species	sex	age	fixed	legs	pounds	weeks
Sasha	cat	female	1	false	4	6.5	3
Snuffles	rabbit	female	3	true	4	3.5	8
Mittens	cat	female	2	true	4	7.4	1
Sunflower	cat	female	5	true	4	8.1	6
Felix	cat	male	16	true	4	9.2	5
Sheba	cat	female	7	true	4	8.4	6
Billie	snail	hermaphrodite	0.5	false	0	0.1	3
Snowcone	cat	female	2	true	4	6.5	5
Wade	cat	male	1	false	4	3.2	1
Hercules	cat	male	3	false	4	13.4	2
Toggle	dog	female	3	true	4	48	1
Boo-boo	dog	male	11	true	4	123	24
Fritz	dog	male	4	true	4	92	3
Midnight	dog	female	5	false	4	112	4
Rex	dog	male	1	false	4	28.9	9
Gir	dog	male	8	false	4	88	5
Max	dog	male	3	false	4	52.8	8
Nori	dog	female	3	true	4	35.3	1
Mr. Peanutbutter	dog	male	10	false	4	161	6
Lucky	dog	male	3	true	3	45.4	9
Kujo	dog	male	8	false	4	172	30
Buddy	lizard	male	2	false	4	0.3	3
Gila	lizard	female	3	true	4	1.2	4
Bo	dog	male	8	true	4	76.1	10
Nibblet	rabbit	male	6	false	4	4.3	2
Snuggles	tarantula	female	2	false	8	0.1	1
Daisy	dog	female	5	true	4	68	8
Ada	dog	female	2	true	4	32	3
Miaulis	cat	male	7	false	4	8.8	4
Heathcliff	cat	male	1	true	4	2.1	2
Tinkles	cat	female	1	true	4	1.7	3
Maple	dog	female	3	true	4	51.6	4

Categorical or Quantitative?

For each piece of data below, circle whether it is **Categorical** or **Quantitative** data.

1	Hair color	categorical	quantitative
2	Age	categorical	quantitative
3	ZIP Code	categorical	quantitative
4	Year	categorical	quantitative
5	Height	categorical	quantitative
6	Sex	categorical	quantitative
7	Street Name	categorical	quantitative

For each question, circle whether it will be answered by **Categorical** or **Quantitative** data.

8	We'd like to find out the average price of cars in a lot.	categorical	quantitative
9	We'd like to find out the most popular color for cars.	categorical	quantitative
10	We'd like to find out which puppy is the youngest.	categorical	quantitative
11	We'd like to find out which cats have been fixed.	categorical	quantitative
12	We want to know which people have a ZIP code of 02907.	categorical	quantitative
13	We'd like to sort a list of phone numbers by area code.	categorical	quantitative

Questions and Column Descriptions

What questions can you ask about the animals dataset? For each question, **can it be answered by this dataset?** Make sure you have at least two questions that can be answered, and at least one that cannot.

What do you NOTICE about this dataset?	What do you WONDER about this dataset?	Answered by this dataset?
		Yes *No*
		Yes *No*
		Yes *No*
		Yes *No*
		Yes *No*
		Yes *No*
		Yes *No*

1. This dataset is Animals that came from an animal shelter, which contains 32 data rows.

2. Some of the columns are:

   a. species, which contains categorical data. Some example values are: "cat", "dog", and "rabbit".

   b. _____, which contains _____ data. Some example values are: _____.

What's on your mind?

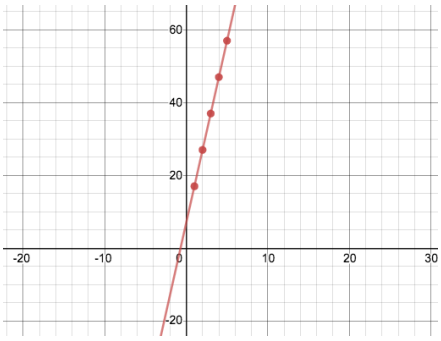
Linear Relationships

A relationship between two variables is **linear** if one changes at a *constant rate* relative to the other. Here are a few examples of linear relationships:

- A car driving at 40mph will travel exactly 40 miles for each additional hour
- A lemonade stand that sells cups of lemonade for $0.75/ea will charge exactly $0.75 for each additional glass

We can see linear relationships show up in **Tables**, **Graphs**, and **Function Definitions**:

x	-1	-2	-3
y	1	2	3



$$f(x) = 10x + 7$$

```
fun f(x): (10 * x) + 7 end
```

In **Graphs**, linear relationships appear as points that form a *straight line*. These lines have a *slope* ("rise over run") and a y-intercept (where the line crosses the y-axis, at x=0).

In **Tables**, linear relationships show up as y-values that change by a *constant rate* relative to their x-values.

We can *define* linear relationships using **Function Definitions** (either in *function notation* or Pyret code). Linear functions always include a term for the slope and another for the y-intercept.

If you know how to read the slope and y-intercept for Tables, Graphs and Definitions, you can switch back and forth between each representation. This flexibility is good: sometimes it's just easier to look at a table or a graph, or see the definition!

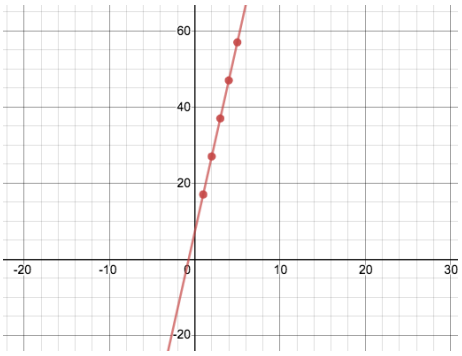
Matching Tables to Graphs

For each of the tables below, find the graph that matches. **Note:** The tables are shown sideways to save space.

x	1	2	3	4	5
y	4	5	6	7	8

1

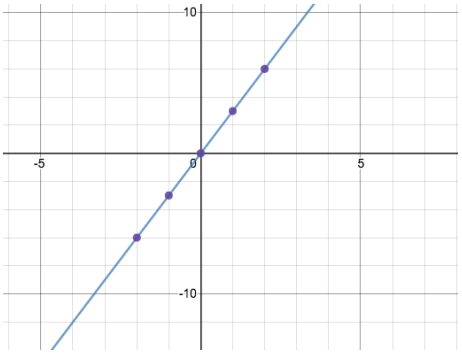
A



x	-5	-4	-3	-2	-1
y	5	4	3	2	1

2

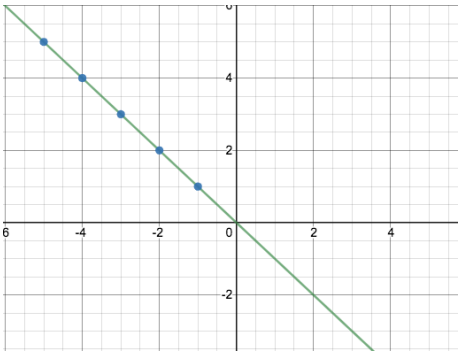
B



x	1	2	3	4	5
y	17	27	37	47	57

3

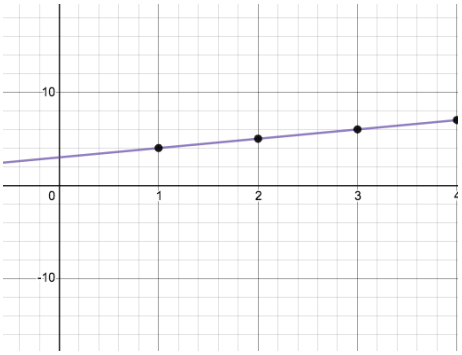
C



x	-2	-1	0	1	2
y	-6	-3	0	3	6

4

D



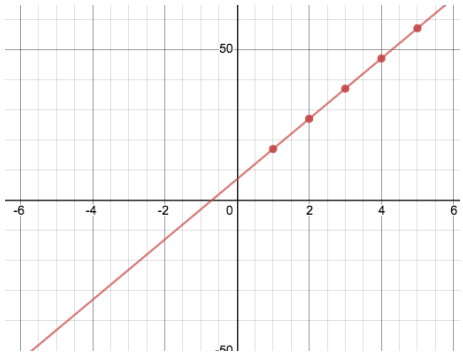
Matching Tables to Graphs 2

For each of the tables below, find the graph that matches. **Note:** The tables are shown sideways to save space, and the graphs have had their scales changed to make the lines *appear* the same. You'll need to look at the axes to find the match!

x	-3	-4	-1	-5	-2
y	3	4	1	5	2

1

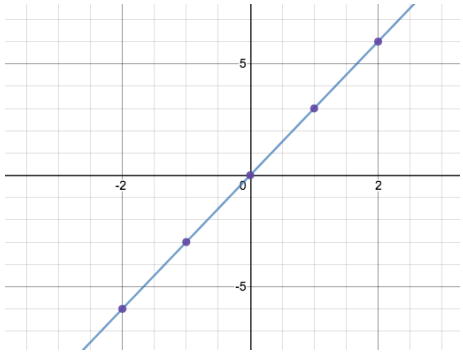
A



x	4	1	3	5	2
y	7	4	6	2	5

2

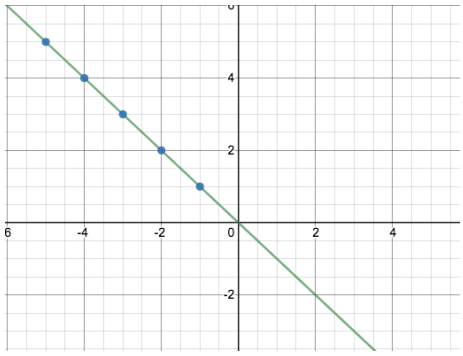
B



x	3	4	5	2	1
y	37	47	57	27	17

3

C



x	3	5	2	1	4
y	9	15	6	3	12

4

D

Linear, Non-linear, or Bust?

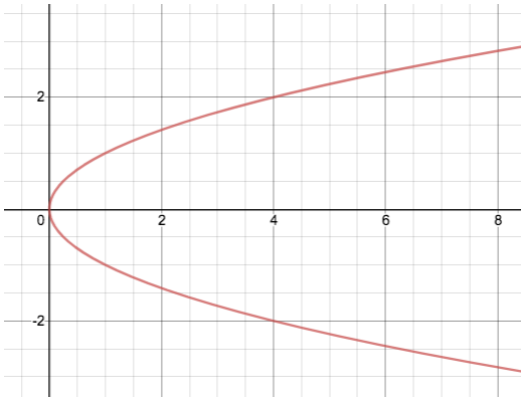
Decide whether each representation is of a linear function, a non-linear function or is not a function at all!

1

x	y
1	5
2	10
3	15
4	20
5	25
6	30
7	35

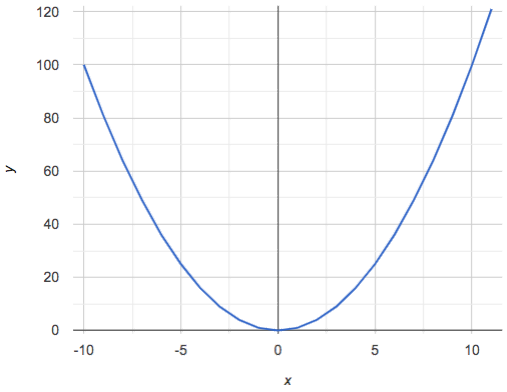
Linear Non-Linear Not a Function

2



Linear Non-Linear Not a Function

3



Linear Non-Linear Not a Function

4

x	y
1	1
2	4
3	9
4	16
5	25
6	37
7	49

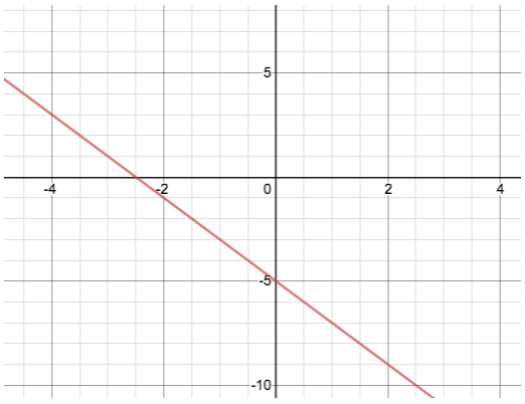
Linear Non-Linear Not a Function

5

x	y
1	1
2	2
3	3
4	4
4	5
6	6
7	9

Linear Non-Linear Not a Function

6



Linear Non-Linear Not a Function

Identifying Slope and y-intercept in Tables

Can you identify the **rate** and **starting value** for the functions represented in each of these tables? Don't forget: *some tables may have their rows out of order!*

1

x	y
0	3
1	5
2	7
3	9

slope/rate: _____

y-intercept: _____

2

x	y
-5	35
-4	28
-3	21
-2	14

slope/rate: _____

y-intercept: _____

3

x	y
12	15
13	15.5
14	16
16	17

slope/rate: _____

y-intercept: _____

4

x	y
1	39
4	36
3	37
2	38

slope/rate: _____

y-intercept: _____

5

x	y
13	57
9	41
11	49
7	33

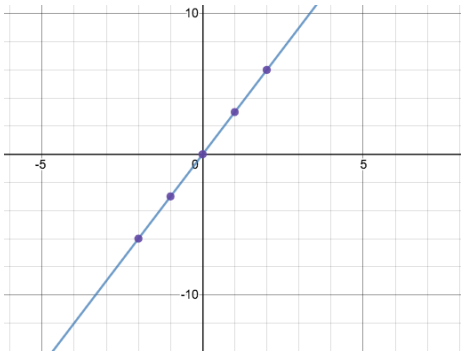
slope/rate: _____

y-intercept: _____

Identifying Slope and y-intercepts in Graphs

Can you identify the **slope** and **y-intercept** for each of these graphs?

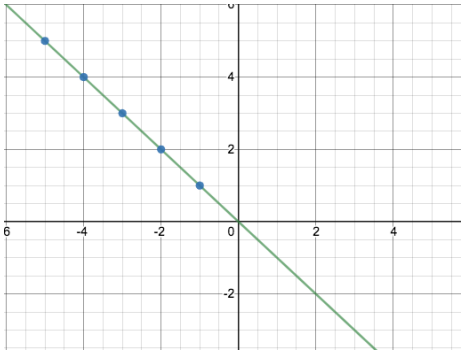
1



slope/rate: _____

y-intercept: _____

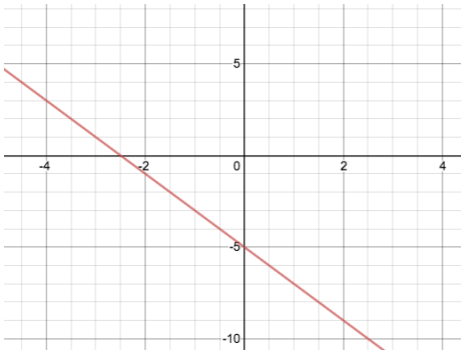
2



slope/rate: _____

y-intercept: _____

3



slope/rate: _____

y-intercept: _____

4

slope/rate: _____

y-intercept: _____

Identifying Slope and y-intercept in Definitions

The following function definitions are written in math notation and in Pyret. Can you identify their **slope** and **y-intercept**?

1	$a(b) = 10b - \frac{2}{5}$		slope: _____ y-intercept: 10
2	`fun c(d) = (7.5 * d) + 22 end`		slope: _____ y-intercept: _____
3	$e(f) = \frac{3}{4}f + 19$		slope: _____ y-intercept: _____
4	`fun g(h): 20 - (16 * h) end`		slope: _____ y-intercept: _____
5	$f(x) = 91 + 4x$		slope: _____ y-intercept: _____
6	`fun i(j): -15 + (1.5 * j) end`		slope: _____ y-intercept: _____

Linear, Non-linear, or Bust?

Decide whether each definition below is a linear function, a non-linear function, or is not a function at all!

1	$boo(x) = 6x - 5$	Linear	Non-Linear	Not a Function
2	$coo(x) = 800 - 9.8x^2$	Linear	Non-Linear	Not a Function
3	`fun doo(x): 10 - (2 * x) end`	Linear	Non-Linear	Not a Function
4	`fun foo(x): (3.75 * x) - 10 end`	Linear	Non-Linear	Not a Function
5	$gloo(x) = 17 - 2.5x$	Linear	Non-Linear	Not a Function
6	$shoo(x) = \sqrt{x}$	Linear	Non-Linear	Not a Function
7	`fun loo(x): 18.6521 end`	Linear	Non-Linear	Not a Function
8	`fun moo(x): (8 * x) end`	Linear	Non-Linear	Not a Function
9	$noo(x) = \frac{-3}{4}x + 100$	Linear	Non-Linear	Not a Function
10	$roo(x) = \sqrt{16}$	Linear	Non-Linear	Not a Function
11	`fun soo(x): 6 / x end`	Linear	Non-Linear	Not a Function
12	`fun too(x): -1.2 * x end`	Linear	Non-Linear	Not a Function
13	$voo(x) = -21x$	Linear	Non-Linear	Not a Function

Matching Graphs to Function Definitions

Match the function definitions to the graphs.

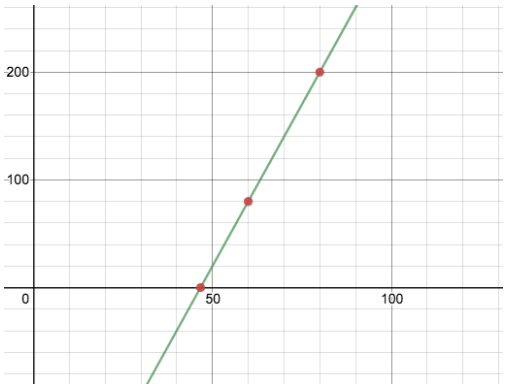
```
fun f(x): (-2/3 * x) + 4 end
```
1

$g(x) = 2x - 10$
2

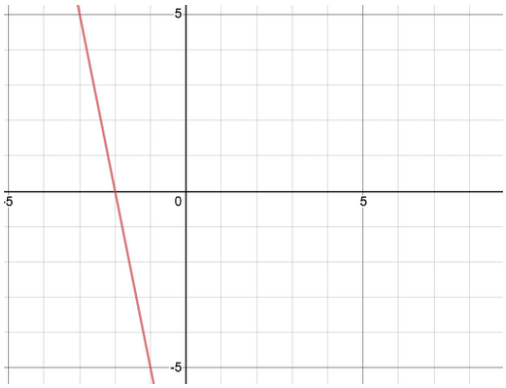
```
fun h(x): (0.5 * x) + 2 end
```
3

$i(x) = 6x + -280$
4

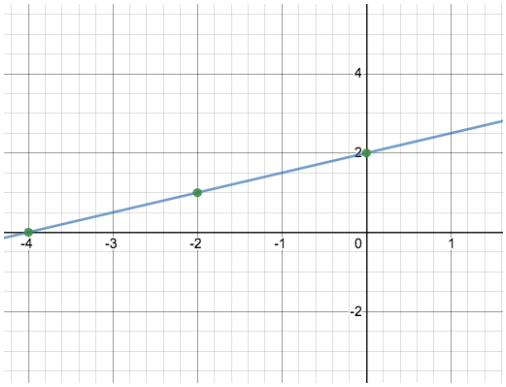
A



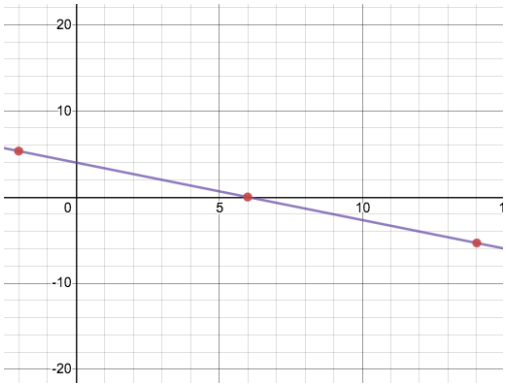
B



C



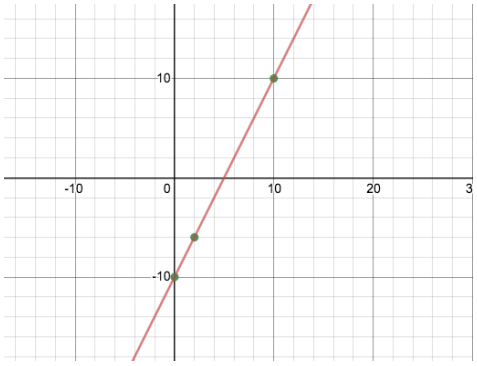
D



Summarizing Graphs with Function Definitions

For each of the Graphs below, write the corresponding function definition, using both Pyret notation *and* function notation. The first one has been done for you.

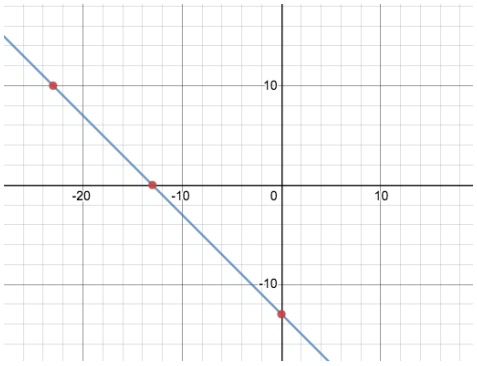
1



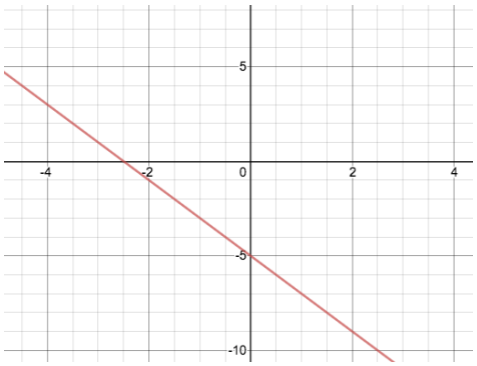

```
fun f(x): (2 * x) - 10 end
```

$$f(x) = 2x - 10$$

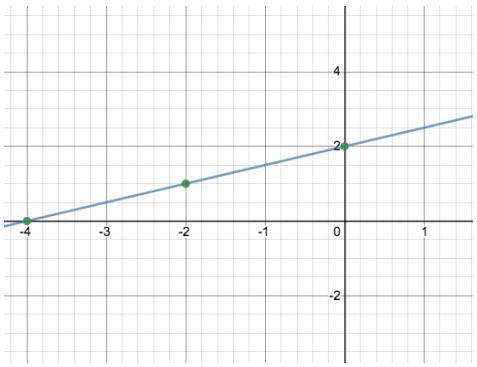
2



3



4



Matching Tables to Function Notation

Match each function definition to the corresponding table.

Note: The tables are shown sideways to save space.

```
fun f(x): (-1 * x) end
```

1

A

x	1	2	3	4	5
y	1	4	9	16	25

```
fun f(x): x + 3 end
```

2

B

x	1	2	3	4	5
y	-1	-2	-3	-4	-5

```
fun f(x): 3 * x end
```

3

C

x	1	2	3	4	5
y	4	5	6	7	8

```
fun f(x): (3 * x) - 5 end
```

4

D

x	-2	-1	0	1	2
y	-11	-8	-5	-2	1

```
fun f(x): num-sqr(x) end
```

5

E

x	1	2	3	4	5
y	3	6	9	12	15

Summarizing Tables with Function Definitions

For each of the Tables below, define corresponding function using Pyret code and function notation. We've started the first function out for you. (**Note:** The tables have been turned on their sides, to save space!)

1

x	0	1	2	3	4
y	-2	0	2	4	6

```
fun f(x):                                    end
```

$f(x) =$

2

x	-2	-1	0	1	2
y	-2	-1	0	1	2

```
fun f(x):                                    end
```

$f(x) =$

3

x	-5	-4	-3	-2	-1
y	9	7	5	3	1

4

x	1	2	3	4	5
y	-1	-2	-3	-4	-5

5

x	9	10	11	12	13
y	14	16	18	20	22

6

x	20	21	22	23	24
y	15	15.5	16	16.5	17

Displaying Categorical Data

Data Scientists use **displays** to visualize data. You've probably seen some of these charts, graphs and plots yourselves! When it comes to displaying **Categorical Data**, there are two displays that are especially useful.

1. **Bar charts** show the *count or percentage* of rows in each category.
    - Bar charts provide a visual representation of the frequency of values in a categorical column.
    - Bar charts have a bar for every category in a column.
    - The more rows in a category, the taller the bar.
    - Bars in a bar chart can be show in *any order*, without changing the meaning of the chart. However, bars are usually shown in some sensible order (bars for the number of orders for different t-shirt sizes might be presented in order of smallest to largest shirt).
2. **Pie charts** show the *percentage* of rows in each category.
    - Pie charts provide a visual representation of the relative frequency of values in a categorical column.
    - Pie charts have a slice for every category in a column.
    - The more rows in a category, the larger the slice.
    - Slices in a pie chart can be shown in *any order*, without changing the meaning of the chart. However, slices are usually shown in some sensible order (e.g. slices might be shown in alphabetical order or from the smallest to largest slice).

Exploring Displays

Using your Contracts page and the Animals Starter File, make each type of display below in pyret. Then sketch the displays and answer the questions. Be sure to add examples of the code you use to your contracts page!

Pie Charts	Bar Charts
Sketch a pie chart here.	Sketch a bar chart here.
Displays *1* column(s) of *categorical* data. What does this display tell us?	Displays ___ column(s) of _____ data. What does this display tell us?
Box Plots	**Histograms**
Sketch a box plot here	Sketch a histogram here
Displays ___ column(s) of _____ data. What do you think this display tells us?	Displays ___ column(s) of _____ data. What do you think this display tells us?

(More) Exploring Displays

For each type of display, fill in the information below.

Scatter Plot	Linear Regression Plot
Sketch a scatter plot here.	Sketch a linear regression plot here.
Displays ___ column(s) of _____ data. What do you think this display tells us? _____ _____ _____ _____	Displays ___ column(s) of _____ data. What do you think this display tells us? _____ _____ _____ _____

What's on your mind?

Data Displays and Lookups

Data scientists use data visualizations to gain better insights into their data, and to communicate their findings with others. Making a display requires answering three questions:

1. **What data** is being displayed? This could be "a random sample of 2000 people", "every animal from the shelter", or "students aged 14-17".
2. **What variables** are being explored? Are we looking at the `species` column? The number of `kilograms` that an animal weighs? Searching for a relationship between a person's `income` and their `height`?
3. **What display** is being used, given the variables being explored? If it's a quantitative variable, we might use a histogram or box plot. If it's categorical, we could use a pie or bar chart. If it's two quantitative variables, we probably want a scatter plot.

Defining Values, Looking up Rows and Columns

We can define names for values in Pyret, the same way we do in math:

```
name = "Flannery"
age = 16
logo = star(50, "solid", "red")
```

When **looking up a data Row** from a Table, programmers use the `row-n` method. This method takes a single number as its input, which tells the computer which Row we want. *Note: Rows are numbered starting at zero!*
For example:

```
sasha   = animals-table.row-n(0)  # define sasha to be the first row
mittens = animals-table.row-n(2)  # define mittens to be the third row
```

When **looking up a column** from a Row, programmers use square brackets and the name of the column they want.
For example:

```
animals-table.row-n(0)["age"]  # look up the age in the 1st row
mittens["species"]             # look up the species in the third row
```

Throughout the rest of the workbook, we will sometimes refer to animalA and animalB as rows from the table.

```
animalA = animals-table.row-n(4)
animalB = animals-table.row-n(13)
```

What Display Goes with Which Data?

Match the Display with the description of the data being plotted. Some descriptions may go with more than one display!

Pie Charts	**1**	**A**	1 column of Quantitative Data
Bar Charts	**2**		
Histograms	**3**	**B**	2 columns of Quantitative Data
Box Plots	**4**		
Scatter Plots	**5**	**C**	1 column of Categorical Data

Data Displays

Fill in the tables below, then use Pyret to make the following displays. Record the code you used.
The first column has been filled in for you.

1) A `pie-chart` showing the `species` of animals from the shelter.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code:

---

2) A `bar-chart` showing the `sex` of animals from the shelter.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code:

---

3) A `histogram` of the number of `pounds` that animals weigh.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code:

---

4) A `box-plot` of the number of `pounds` that animals weigh.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code:

---

5) A `scatter-plot`, using the animals' `species` as the labels, `age` as the x-axis, and `pounds` as the y-axis.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code:

---

6) A `scatter-plot`, using the animals' `name` as the labels, `pounds` as the x-axis, and `weeks` as the y-axis.

Which Rows?	Which Column(s)?	What Display?
All the animals		

code:

Lookup Questions

The table below represents four pets:

`pets-table`

name	sex	age	pounds
`"Toggle"`	`"female"`	`3`	`48`
`"Fritz"`	`"male"`	`4`	`92`
`"Nori"`	`"female"`	`6`	`35.3`
`"Maple"`	`"female"`	`3`	`51.6`

1) *Match* each Lookup Question (left) to the code that will give the answer (right).

"How much does Maple weigh?"	**1**	**A**	`pets-table.row-n(3)`
"Which is the last row in the table?	**2**	**B**	`pets-table.row-n(2)["name"]`
"What is Fritz's sex?"	**3**	**C**	`pets-table.row-n(1)["sex"]`
"What's the third animal's name?"	**4**	**D**	`pets-table.row-n(3)["age"]`
"How much does Nori weigh?"	**5**	**E**	`pets-table.row-n(3)["pounds"]`
"How old is Maple?"	**6**	**F**	`pets-table.row-n(0)`
"What is Toggle's sex?"	**7**	**G**	`pets-table.row-n(2)["pounds"]`
"What is the first row in the table?"	**8**	**H**	`pets-table.row-n(0)["sex"]`

2) Fill in the blanks (left) with code that will produce the value (right).

a.	pets-table.row-n(3)["name"]	`"Maple"`
b.	_____	`"male"`
c.	_____	`4`
d.	_____	`48`
e.	_____	`"Nori"`

What's on your mind?

Defining Row Functions & Using Table Methods

Methods are special functions that are attached to pieces of data. We use them to manipulate Tables.

- In this course, the methods we'll be using are
  - `row-n` - consumes an index (starting with zero!) and produces a row from a table
  - `order-by` - consumes the name of a column and a Boolean value to determine if that table should be sorted by that column in ascending order
  - `filter` - consumes a *Boolean-producing function*, and produces a table containing only rows for which the function returns `true`
  - `build-column` - consumes the name of a new column, and a function that produces the values in that column for each Row
- Unlike functions, methods can't be used alone. They have a "secret" argument, which is the data they are attached to. They are written as part of that data, separated by a dot. For example:

  ```
  shapes.row-n(2)
  ```
- Contracts for methods are different from other functions. They include the type of the data as part of their names. For example:

  ```
  <table>.row-n :: (index :: Number) -> Row
  ```

Reading Row and Function Definitions

Make sure you've opened the Table Methods Starter File on your computer.

1	What name is being defined on line 15?	
2	How many columns are listed here?	
3	What name is being defined on line 22?	
4	Is `cat-row` a Number, String, Image or Row?	
5	Type `cat-row` into the Interactions Area. What do you get?	

6) On line 27, define `dog-row`. After clicking "Run", type `dog-row` into the Interactions Area and make sure it's a dog! Do the same for `old-row` and `unfixed-row`.

7	A Contract for a function is written on line 39. What is its name?	
8	What is its Domain?	
9	What is its Range?	
10	What other functions are defined here?	

11) Lines 41-42 **define a new function!** What does this function do?

Exploring Row and Function Definitions

Make sure you've opened the Table Methods Starter File on your computer.

1	Evaluate `is-dog(dog-row)`. What do you get?	
2	Evaluate `is-cat(cat-row)`. What do you get?	
3	Evaluate `is-cat(dog-row)`. What do you get?	
4	Evaluate `is-dog(dog-row)`. What do you get?	
5	Evaluate `is-dog(cat-row)`. What do you get?	
6	What does `is-cat` do?	
7	What does `lookup-fixed` do?	
8	What does `is-old` do?	
9	What does `kilos` do?	
10	What does `nametag` do?	

11) Find the Contract for `image-scatter-plot` in your Contracts page, and discuss the Domain as a group.

12) In the Interactions Area, type `image-scatter-plot(animals-table, "pounds", "weeks", nametag)`. What do you get?

---

13) Change the definition of `nametag` to produce text with a different color.

14) Change the definition of `nametag` to produce text with a different size.

15) Change the definition of `nametag` to produce text using the animal's `species`, instead of their `name`.

16) Change the definition of `nametag` to produce text using the animal's `age` as the size of the text.

The Design Recipe

Functions have multiple representations (e.g. - Contracts, Examples, and Definition), and each of these representations shows us a particular part of how the function should behave. By using these representations in a particular order - called the ***Design Recipe*** - we can build lots of functions, check our work, and document our thinking!

Contract and Purpose Statement

The first step in the Design Recipe is to write the Contract. This means we have to be able to answer three questions:

- What is the **Name** of the function we are defining?
- What is the **Domain** of that function? (When dealing with Table Functions, the Domain is always `Row`)
- What is the **Range** of the function? (What is the type of the output?)

The Purpose Statement is a way of adding detail to the Contract, using plain human language. A good Purpose Statement should always explain:

- What the input *represents*. (Is it Animals? Schools? States?)
- What the output *represents*. (Pounds? True or false?)
- All the information necessary to go from input to output.

It's important to start with this representation, because it's the least detailed. If we can't answer *these* questions, we shouldn't start writing code!

Examples

The second step is work through some concrete examples, making sure that we know exactly what the function will do.
The goal of the Examples step is to *find the pattern* that represents what the function does. Sometimes we have to start by just focusing on what the answer should be. Suppose `animalA` is a lizard animal, and `animalB` isn't. We can imagine the answers for an `is-lizard` to be...

```
examples:
    is-lizard(animalA) is true
    is-lizard(animalB) is false
end
```

But what work do we have to do to check if an animal is a lizard? (1) We **look up** the "species" column, and (2) ask if the value is equal to "lizard". We can write both of these steps in code, finishing the examples:

```
examples:
    is-lizard(animalA) is animalA["species"] == "lizard"
    is-lizard(animalB) is animalB["species"] == "lizard"
end
```

(And sometimes we can go straight to showing our work, doing the whole thing in one step!)
Once we see the pattern, we can *circle and label what changes*. In this case, only the animal itself changes!

Definition

The final step in the Design Recipe is to take the pattern from our examples and *generalize it* to work with any input.
Once again, our previous step is a huge help: we can simply **copy everything that stays the same**, and replace the part that changes with the label we used:

```
fun is-lizard(r): r["species"] == "lizard" end
```

The Design Recipe - Compute

For the word problems below, assume `dog-row`, `cat-row`, `young-row` and `old-row` are already defined as data rows.

Directions: Define a function called `is-cat`, which consumes a `Row` of the animals table and *computes* whether the animal is a cat.

Contract and Purpose Statement

Every contract has three parts…

```
# is-cat:: Row -> Boolean
```
function name | *domain* | *range*

```
# Consumes an animal, and computes whether the species equals "cat"
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

```
is-cat ( dog-row ) 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 is-cat( r ):
  r["species"] == "cat"
```
function name | *variable(s)* | *what the function does with those variable(s)*

end

Directions: Define a function called `is-young`, which consumes a Row of the animals table and *computes* whether it is less than four years old.

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

The Design Recipe - Lookup

For the word problems below, assume `fixed` and `unfixed` are already defined as data rows.

Directions: Define a function called `lookup-fixed`, which looks up whether or not an animal is fixed.

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:

`lookup-fixed ( fixed-row ) is fixed-row["fixed"]`
function name *input(s)* *what the function produces*

`lookup-fixed ( unfixed-row ) is unfixed-row["fixed"]`
function name *input(s)* *what the function produces*

end

Definition

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

fun `lookup-fixed( r ):`
function name *variable(s)*

`r["fixed"]`
what the function does with those variable(s)

end

Directions: Define a function called `lookup-name`, which consumes a Row of the animals table and looks up the name of that animal.

Contract and Purpose Statement

Every contract has three parts…

`lookup-name`:: `Row` -> `String`
function name *domain* *range*

Consumes an animal, and looks up the name
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

What's on your mind?

Method Chaining

Method chaining allows us to apply multiple methods with less code.

For example, instead of using multiple definitions, like this:

```
with-labels = animals-table.build-column("labels", nametag)
cats = with-labels.filter(is-cat)
cats.order-by("age", true)
```

We can use method-chaining to write it all on one line, like this:

```
animals-table.build-column("labels", nametag).filter(is-cat).order-by("age", true)
```

Order Matters! The methods are applied in the order they appear. For example, trying to order a table by a column that hasn't been built will result in an error.

The Design Recipe

For the word problems below, assume you have `animalA` and `animalB` defined in your code.

Directions: Define a function called `is-dog`, which consumes a `Row` of the animals table and *computes* whether the animal is a dog.

Contract and Purpose Statement

Every contract has three parts…

```
# is-dog:: Row -> Boolean
```
function name *domain* *range*

```
# Consumes an animal, and computes whether the species == "dog"
```
what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

```
is-dog ( animalA ) is animalA["species"] == "dog"
```
function name *input(s)* *what the function produces*

```
is-dog ( animalB ) is
```
function name *input(s)* *what the function produces*

end

Definition

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

```
fun is-dog( r ):
```
function name *variable(s)*

```
  r["species"] == "dog"
```
what the function does with those variable(s)

end

Directions: Define a function called `is-female`, which consumes a Row of the animals table and returns true if the animal is female.

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

The Design Recipe

For the word problems below, assume you have `animalA` and `animalB` defined in your code.

Directions: Define a function called `is-old`, which consumes a Row of the animals table and *computes* whether it is more than 12 years old.

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: Define a function called `name-has-s`, which returns true if an animal's name contains the letter "s"

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 name-has-s(r):
function name *variable(s)*

```
string-contains(r["name"], "s")
```
what the function does with those variable(s)

end

Chaining Methods

You have the following functions defined below (read them *carefully!*):

```
fun is-fixed(r): r["fixed"]                end
fun is-young(r): r["age"] < 4              end
fun nametag(r):  text(r["name"], 20, "red") end
```

The table `t` below represents four animals from the shelter:

name	sex	age	fixed	pounds
`"Toggle"`	`"female"`	`3`	`true`	`48`
`"Fritz"`	`"male"`	`4`	`true`	`92`
`"Nori"`	`"female"`	`6`	`true`	`35.3`
`"Maple"`	`"female"`	`3`	`true`	`51.6`

Match each Pyret expression (left) to the description of what it does (right).

`t.order-by("age", true)`	1	A	Produces a table containing only Toggle and Maple
`t.filter(is-fixed)`	2	B	Produces a table of only young, fixed animals
`t.build-column("sticker", nametag)`	3	C	Produces a table, sorted youngest-to-oldest
`t.filter(is-young)`	4	D	Produces a table with an extra column, named "sticker"
`t.filter(is-young)` `  .filter(is-fixed)`	5	E	Produces a table containing Maple and Toggle, in that order
`t.filter(is-young)` `  .order-by("pounds", false)`	6	F	Produces a table containing the same four animals
`t.build-column("label", nametag)` `  .order-by("age", true)`	7	G	Won't run: will produce an error
`t.order-by("sx", false)`	8	H	Produces a table with an extra "label" column, sorted youngest-to-oldest

Chaining Methods 2: Order Matters

You have the following functions defined below (read them *carefully!*):

```
fun is-female(r): r["sex"] == "female"  end
fun kilograms(r): r["pounds"] / 2.2     end
fun is-heavy(r):  r["kilos"] > 25       end
```

The table `t` below represents four animals from the shelter:

name	sex	age	fixed	pounds
`"Toggle"`	`"female"`	`3`	`true`	`48`
`"Fritz"`	`"male"`	`4`	`true`	`92`
`"Nori"`	`"female"`	`6`	`true`	`35.3`
`"Maple"`	`"female"`	`3`	`true`	`51.6`

Match each Pyret expression (left) to the description of what it does (right). **Note: one description might match multiple expressions!**

```
t.order-by("kilos", true)
```

1

A Produces a table containing Toggle, Nori and Maple, with an extra column showing their weight in kilograms

```
t.filter(is-female)
  .build-column("kilos", kilograms)
```

2

B Produces a table containing Maple, Nori and Toggle (in that order)

```
t.build-column("kilos", kilograms)
  .filter(is-heavy)
```

3

C Produces a table containing only Fritz.

```
t.filter(is-heavy)
  .build-column("kilos", kilograms)
```

4

D Won't run: will produce an error

```
t.build-column("kilos", kilograms)
  .filter(is-heavy)
  .order-by("sex", true)
```

5

E Produces a table containing only Fritz, with two extra columns.

```
t.build-column("female", is-female)
  .build-column("kilos", kilograms)
  .filter(is-heavy)
```

6

F Produces a table containing Maple and Fritz

What's on your mind?

Randomness and Sample Size

Computer Scientists may take **samples** that are subsets of a data set. If their sample is well chosen, they can use it to test if their code does what it's supposed to do. However, choosing a good sample can be tricky!

Random Samples are a subset of a population in which each member of the subset has an equal chance of being chosen. A random sample is intended to be a representative subset of the population. The larger the random sample, the more closely it will represent the population and the better our inferences about the population will tend to be.

Grouped Samples are a subset of a population in which each member of the subset was chosen for a specific reason. For example, we might want to look at the difference in trends between two groups ("Is the age of a dog a bigger factor in adoption time v. the age of a cat?"). This would require making grouped samples of *just the dogs* and *just the cats*.

Sampling and Inference

1) Evaluate the `big-animals-table` in the Interactions Area. This is the *complete* population of animals from the shelter! Below is a true statement about that population:

> The population is 47.7% fixed and 52.3% unfixed.

Type each of the following lines into the Interactions Area and hit "Enter".

```
random-rows(big-animals-table, 10)
random-rows(big-animals-table, 40)
```

2) What do you get?

3) What is the contract for `random-rows` ?

4) What does the `random-rows` function do?

5) In the Definitions Area, define `small-sample` and `large-sample` to be these two random samples.

6) Make a `pie-chart` for the animals in each sample, showing percentages of fixed and unfixed.

- The percentage of fixed animals in the entire populations is **47.7%** .
- The percentage of fixed animals in `small-sample` is _____ .
- The percentage of fixed animals in `large-sample` is _____ .

7) Make a `pie-chart` for the animals in each sample, showing percentages for each species.

- The percentage of tarantulas in the entire population is **roughly 5%** .
- The percentage of tarantulas in `small-sample` is _____ .
- The percentage of tarantulas in `large-sample` is _____ .

8) Click "Run" to direct the computer to generate a different set of random samples of these sizes. Make a new `pie-chart` for each sample, showing percentages for each species.

- The percentage of tarantulas in the entire population is **roughly 5%** .
- The percentage of tarantulas in `small-sample` is _____ .
- The percentage of tarantulas in `large-sample` is _____ .

9) Which repeated sample gave us a more accurate inference about the whole population? Why?

Grouped Samples from the Animals Dataset

Use method chaining to define the **grouped samples** below, using the helper functions that you've already defined: `is-old`, `is-young`, `is-cat`, `is-dog`, `is-female`, `lookup-fixed`, and `has-s-name`. We've given you the solution for the first sample, to get you started.

	Subset	The code to define that subset
1	Kittens	`kittens = animals-table.filter(is-cat).filter(is-young)`
2	Puppies	
3	Fixed Cats	
4	Cats with "s" in their name	
5	Old Dogs	
6	Fixed Animals	
7	Old Female Cats	
8	Fixed Kittens	
9	Fixed Female Dogs	
10	Old Fixed Female Cats	

Displaying Data

Fill in the tables below, then use Pyret to make the following displays. Record the code you used. The first table has been filled in for you.

1) A `bar-chart` showing how many puppies are fixed or not.

What Rows?	Which Column(s)?	What Display?
puppies	*fixed*	*bar-chart*

code: `bar-chart(animals-table.filter(is-dog).filter(is-young), "fixed")`

2) A `pie-chart` showing how many heavy dogs are fixed or not.

What Rows?	Which Column(s)?	What Display?

code: _____

3) A `histogram` of the number of `weeks` it takes for a random sample of animals to be adopted.

What Rows?	Which Column(s)?	What Display?

code: _____

4) A `box-plot` of the number of `pounds` that kittens weigh.

What Rows?	Which Column(s)?	What Display?

code: _____

5) A `scatter-plot` of a random sample using `species` as the labels, `age` as the x-axis, and `weeks` as the y-axis.

What Rows?	Which Column(s)?	What Display?

code: _____

6) Describe **your own grouped sample** here, and fill in the table below.

_____

What Rows?	Which Column(s)?	What Display?

code: _____

What's on your mind?

Histograms

To best understand histograms, it's helpful to contrast them first with bar charts.

Bar charts show the number of rows belonging to a given category. The more rows in each category, the taller the bar.

- *Bar charts provide a visual representation of the frequency of values in a* ***categorical*** *column.*
- There's no strict numerical way to order these bars, but **sometimes there's an order** that makes sense. For example, bars for the sales of different t-shirt sizes might be presented in order of smallest to largest shirt.

Histograms show the number of rows that fall within certain intervals, or "bins", on a horizontal axis. The more rows that fall within a particular "bin", the taller the bar.

- *Histograms provide a visual representation of the frequencies (or relative frequencies) of values in a* ***quantitative*** *column.*
- Quantitative data **can always be ordered**, so the bars of a histogram always progress from smallest (on the left) to largest (on the right).
- When dealing with histograms, it's important to select a good **bin size**. If the bins are too small or too large, it is difficult to see the shape of the dataset. Choosing a good bin size can take some trial and error!

The **shape** of a data set tells us which values are more or less common.

- In a **symmetric** data set, values are just as likely to occur a certain distance above the mean as below the mean.
- A data set that is **skewed left** and/or has low outliers has a few values that are unusually low. The histogram for a skewed left dataset has a few data points that are stretched out to the left (lower) end of the x-axis.
- A data set that is **skewed right** and/or high outliers means there are a few values that are unusually high. The histogram for a skewed right dataset has a few data points that are stretched out to the right (higher) end of the x-axis.
- One way to visualize the difference between a histogram of data that is **skewed left** or **skewed right** is to think about the lengths of our toes on our left and right feet. Much like a histogram that is "skewed left", our left feet have smaller toes on the left and a bigger toe on the right. Our right feet have the big toe on the left and smaller toes on the right, more closely resembling the shape of a histogram of "skewed right" data.

The Design Recipe

For the word problems below, assume you have `animalA` and `animalB` defined in your code.

Directions: Define a function called `kilos`, which consumes a Row of the animals table and divides the pounds column by 2.2 to *compute* the animal's weight in kilograms.

Contract and Purpose Statement

Every contract has three parts…

_____ :: _____ (r :: Row) _____ -> _____

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: Define a function called `smart-dot`, which consumes a Row of the animals table and *computes* the image of a solid red circle using the animal's `pounds` as the radius.

Contract and Purpose Statement

Every contract has three parts…

smart-dot:: _____ -> Image

function name *domain* *range*

Consumes an animal, and computes a solid red circle using the weight in pounds as the radius

what does the function do?

Examples

Write some examples, then circle and label what changes…

examples:

smart-dot ("animalA") **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)

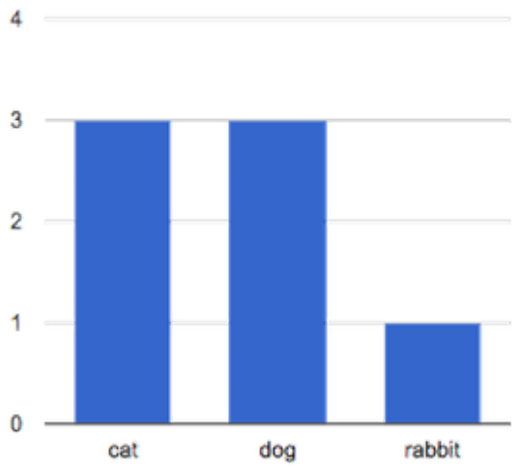
end

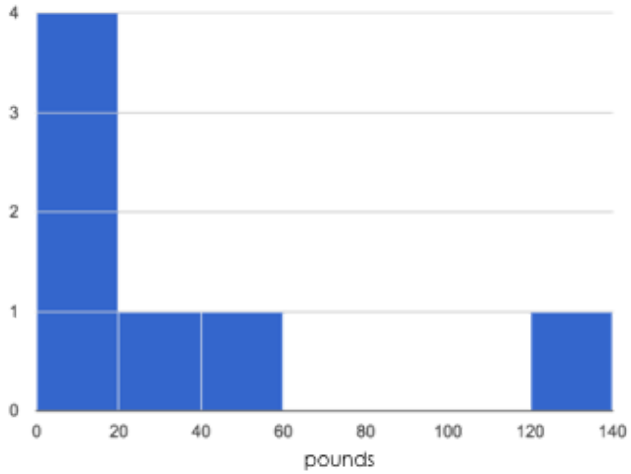
Summarizing Columns

name	species	age	pounds
"Sasha"	"cat"	1	6.5
"Boo-boo"	"dog"	11	12.3
"Felix"	"cat"	16	9.2
"Nori"	"dog"	6	35.3
"Wade"	"cat"	1	3.2
"Nibblet"	"rabbit"	6	4.3
"Maple"	"dog"	3	51.6

1	How many cats are there in the table above?	
2	How many dogs are there?	
3	How many animals weigh between 0-20 pounds?	
4	How many animals weigh between 20-40 pounds?	
5	Are there more animals weighing 40-60 than 60-140 pounds?	

The charts below are both based on this table. What is similar about them? What is different?





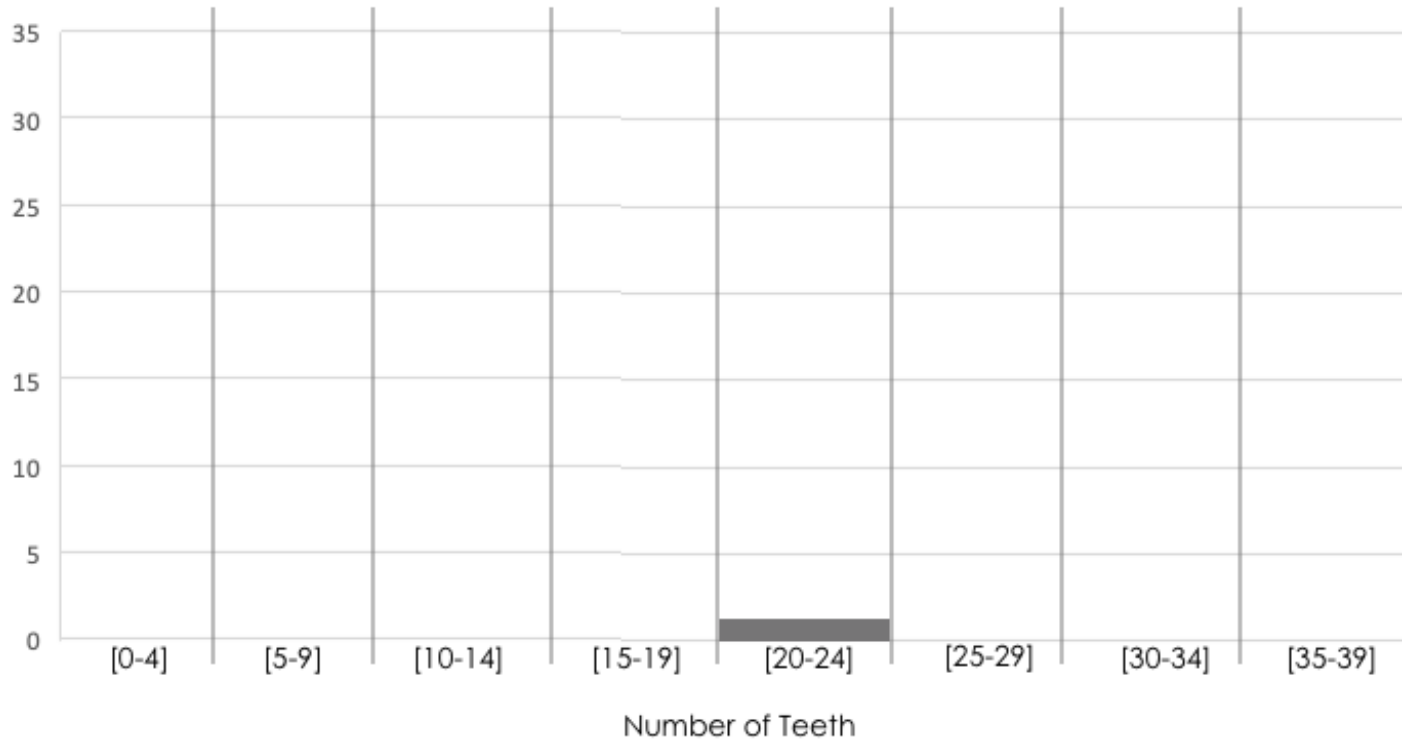
Similarities	Differences

Making Histograms

Suppose we have a data set for a group of 50 adults, showing the number of teeth each person has:

Number of teeth	Count
0	5
22	1
26	1
27	1
28	4
29	3
30	5
31	3
32	27

Draw a histogram for the table in the space below. For each row, find which interval (or "bin") on the x-axis represents the right number of teeth. Then fill in the box so that the height of the box is equal to the *sum of the counts* that fit into that interval. One of the intervals has been completed for you.



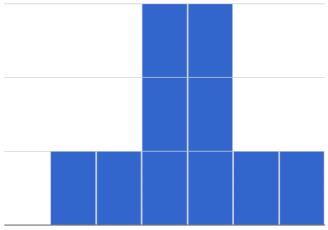
Reading Histograms

Students watched 5 videos, and rated them on a scale of 1 to 10. While the **average score** for every video is the same (5.5), the **shapes** of the ratings distributions were very different! *Match* the summary description (left) with the *shape* of the histogram of student ratings (right). For each histogram, **the x-axis is the score, and the y-axis is the number of students who gave it that score**. These axes are intentionally unlabeled - focusing on the *shape* is what matters here!

Most of the students were fine with the video, but a couple of them gave it an unusually low rating.

1

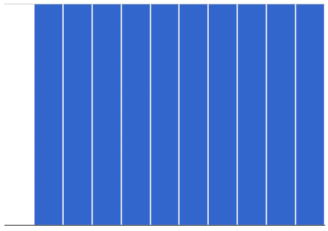
A



Most of the students were okay with the video, but a couple students gave it an unusually high rating.

2

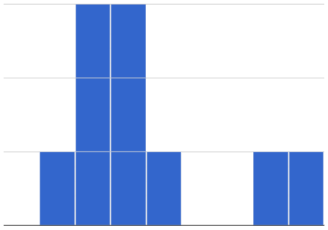
B



Students tended to give the video an average rating, and they weren't likely to stray far from the average.

3

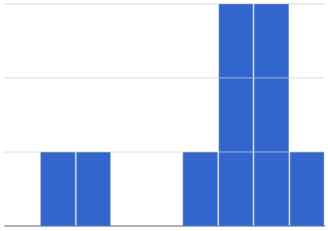
C



Students either really liked or really disliked the video.

4

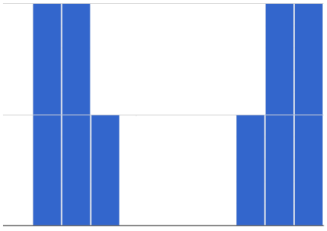
D



Reactions to the video were all over the place: high ratings and low ratings and inbetween ratings were all equally likely.

5

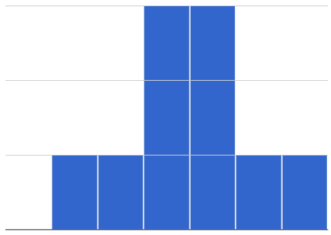
E



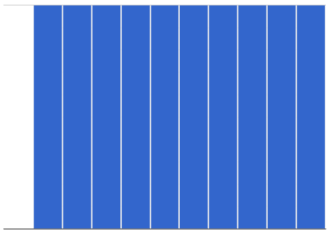
Identifying Shape - Histograms

Describe the shape of histograms on the left in complete sentences, using vocabulary like "Skewed Left", "Skewed Right", or "Symmetric".

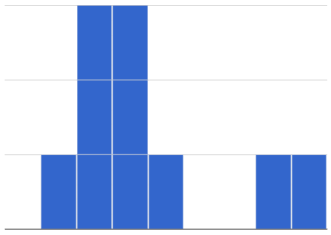
1



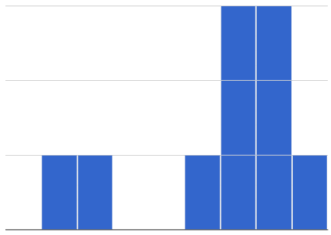
2



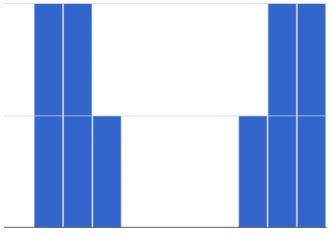
3



4



5



The Shape of the Animals Dataset

Describe two histograms made from columns of the animals dataset.

1) Make a histogram, showing the distribution of pounds (column in your dataset) for animals from the shelter (your subset, e.g., "fixed dogs from the shelter").

2) Make another histogram, showing the distribution of _____ (column in your dataset) for _____ (your subset, e.g., "fixed dogs from the shelter").

3) How would you describe the shape of these histograms?

What do you NOTICE?	What do you WONDER?

The Spread of My Dataset

Describe two of the histograms you made from your dataset.

1) I made a histogram, showing the distribution of _____ for
column in your dataset

_____.
your subset, e.g., "fixed dogs from the shelter"

2) I made a histogram, showing the distribution of _____ for
column in your dataset

_____.
your subset, e.g., "fixed dogs from the shelter"

3) How would you describe the shape of these histograms?

_____

What do you NOTICE about these displays?	What do you WONDER about these displays?

What's on your mind?

Measures of Center and Spread

There are three ways to measure the **center** of a dataset, to summarize a whole column of quantitative data using just one number:

- The **mean** of a dataset is the average of all the numbers.
- The **median** of a dataset is a value that is smaller than half the dataset, and larger than the other half. In an ordered list the median will either be the middle number or the average of the two middle numbers.
- The **mode(s)** of a data set is the value (or values) occurring most often. When all of the values occur equally often, a dataset has no mode.

In a **symmetric** dataset, values are just as likely to occur a certain distance above the mean as below the mean, and the median and mean are usually close together.

When a dataset is asymmetric, the median is a more decriptive measure of center than the median.

- A dataset with **left skew**, and/or low outliers, has a few values that are unusually low, pulling the mean *below* the median.
- A dataset with **right skew**, and/or high outliers, means there are a few values that are unusually high, pulling the mean *above* the median.

When a dataset contains a small number of values, the mode may be the most descriptive measure of center.

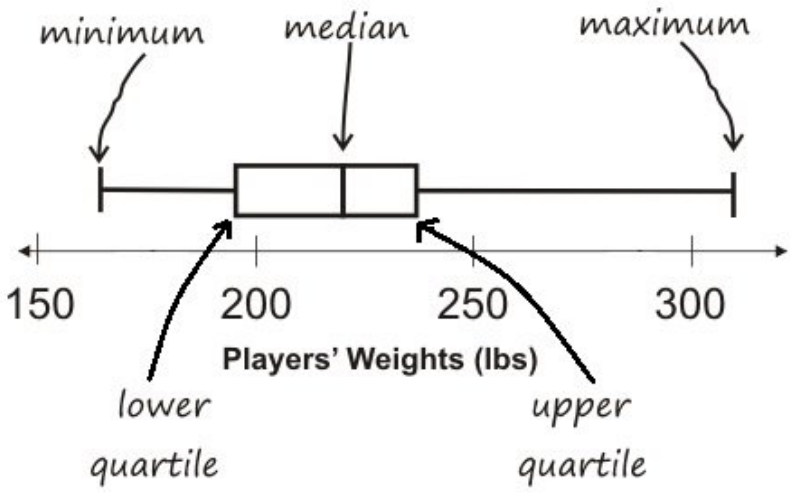
Data Scientists can also measure the **spread** of a dataset using a **five-number summary**:

- The **minimum** – the lowest value in the dataset
- The **first, or "lower" quartile (Q1)** – the middle of the lower half of values, which separates the lowest quarter from the next smallest quarter
- The **second quartile (Q2)** – the middle value, which separates the entire dataset into "top" and "bottom" halves
- The **third, or "upper" quartile (Q3)** – the middle of the higher half of values which separates the second highest quarter from the highest quarter
- The **maximum** – the largest value in the dataset

Measures of Center and Spread (continued)

The **five-number summary** can be used to draw a **box plot**.

- Each of the four sections of the box plot contains 25% of the data. *If the values are distributed evenly across the range, the four sections of the box plot will be equal in width.* Uneven distributions will show up as differently-sized sections of a box plot.
- The left **whisker** extends from the minimum to Q1.
- The **box**, or **interquartile range**, extends from Q1 to Q3. It is divided into 2 parts by the **median**. Each of those parts contains 25% of the data, so the whole box contains the central 50% of the data.
- The right **whisker** extends from Q3 to the maximum.



The box plot above, for example, tells us that:

- The minimum weight is about 165 pounds. The median weight is about 220 pounds. The maximum weight is about 310 pounds.
  - 1/4 of the players weigh roughly between 165 and 195 pounds
  - 1/4 of the players weigh roughly between 195 and 220 pounds
  - 1/4 of the players weigh roughly between 220 and 235 pounds
  - 1/4 of the players weigh roughly between 235 and 310 pounds
  - 50% of the players weigh roughly between 165 and 220 pounds
  - 50% of the players weigh roughly between 195 and 235 pounds
  - 50% of the players weigh roughly between 220 and 310 pounds
- The densest concentration of players' weights is between 220 and 235 pounds.
- Because the widest section of the box plot is between 235 and 310 pounds, we understand that the weights of the heaviest 25% fall across a wider span than the others. 310 may be an outlier, the weights of the players weighing between 235 pounds and 310 pound could be evenly distributed across the range, or all of the players weighing over 235 pounds may weigh around 310 pounds.

Summarizing Columns in the Animals Dataset

Find the measures of center and spread to summarize the pounds column of the Animals Table.

Be sure to add examples to your Contracts page as you work.

Measures of Center

The three measures of center for this column are:

Mean (Average)	Median	Mode(s)

Since the mean is _____ compared to the median, this suggests the shape is

[higher/lower/about equal]

_____.

[skewed right (or high outliers) / skewed left (or low outliers) / symmetric]

Measures of Spread

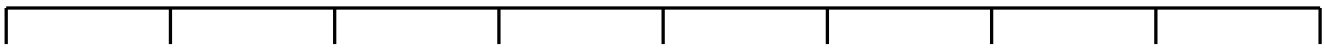
My five-number summary is:

Minimum	Q1	Median	Q3	Maximum

Displaying Center and Spread with a Box Plot

Draw a box plot from this summary on the number line below.

Be sure to label the number line with consistent intervals.



From this summary and box plot, I conclude:

_____

_____

_____

_____

Interpreting Spread

Consider the following dataset, representing the annual income of ten people.
All numbers represent *thousands of dollars* (so `14` means "$14,000"):

```
60, 10, 21, 180, 14, 20, 45, 35, 45, 170
```

1) In the space below, rewrite this dataset in **sorted order**.

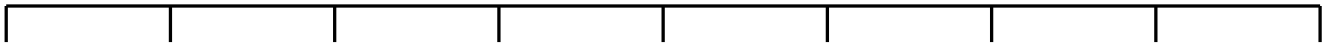
2) In the table below, compute the **measures of center** for this dataset.

Mean (Average)	Median	Mode(s)

3) In the table below, compute the **five number summary** of this dataset.

Minimum	Q1	Q2 (Median)	Q3	Maximum

4) On the number line below, draw a **box plot** for this dataset.

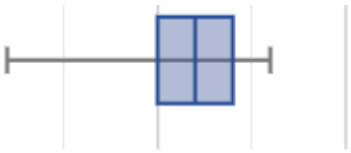
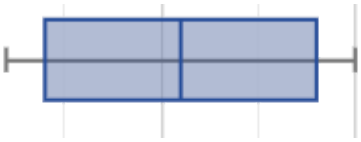
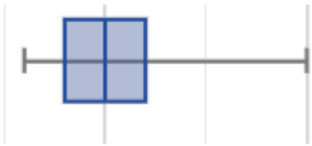
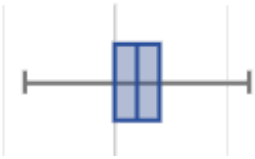
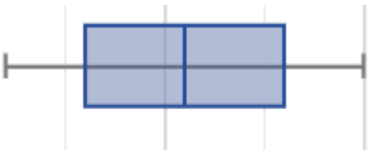


5) The following statements are *correct* … but misleading. Write down the reason why.

Statement	Why it's misleading
"They're rich! The average person makes $60k dollars!"	
"It's a middle-income list: the most common salary is $45k/yr!"	
"This group is very low-income, the most common salary range is from $10k-$25k!"	

Identifying Shape - Box Plots

Describe the shape of box plots on the left in complete sentences, using vocabulary like "Skewed Left", "Skewed Right" or "Symmetric".

1		
2		
3		
4		
5		

Shape of My Dataset

Find the measures of center and spread to summarize a column of your dataset.

The column I chose to summarize is _____.

Measures of Center

The three measures of center for this column are:

Mean (Average)	Median	Mode(s)

Since the mean is _____ compared to the median, this suggests the shape is
[higher/lower/about equal]

_____.
[skewed right (or high outliers) / skewed left (or low outliers) / symmetric]

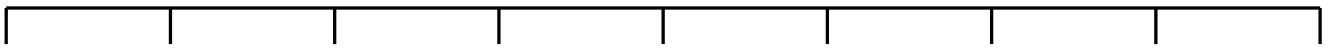
Measures of Spread

My five-number summary is:

Minimum	Q1	Q2 (Median)	Q3	Maximum

Displaying Center and Spread with a Box Plot

Draw a box plot from this summary on the number line below.
Be sure to label the number line with consistent intervals.



From this summary and box plot, I conclude:

_____

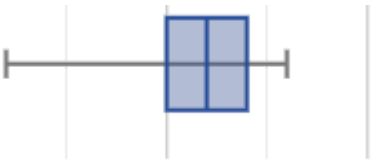
_____

_____

_____

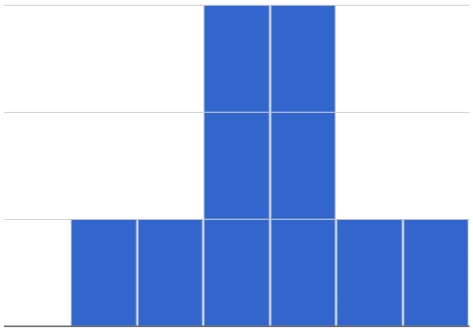
Matching Box-Plots to Histograms

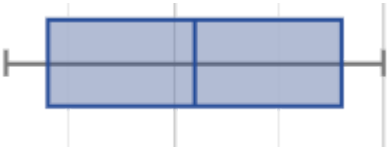
Students watched 5 videos, and rated them on a scale of 1 to 10. For each video, their ratings were used to generate box-plots and histograms. Match the box-plot to the histogram that displays the same data.*



1

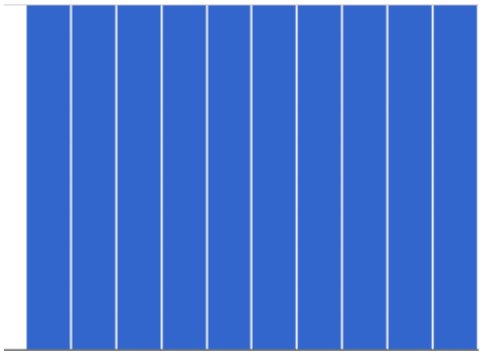
A

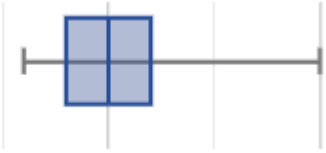




2

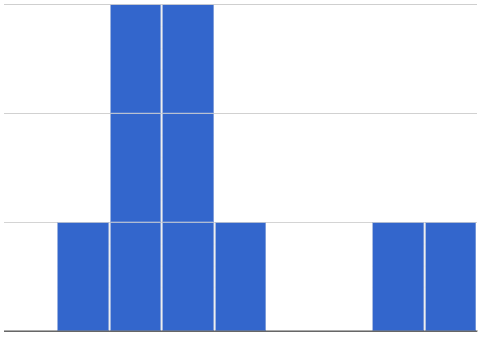
B

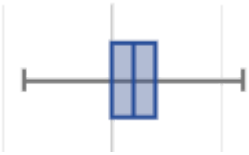




3

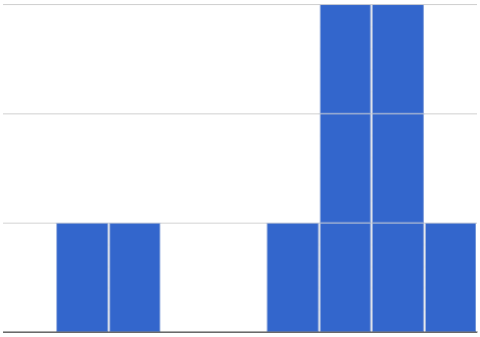
C





4

D



What's on your mind?

Scatter Plots

Scatter Plots can be used to show a relationship between two quantitative columns. Each row in the dataset is represented by a point, with one column providing the x-value and the other providing the y-value. The resulting "point cloud" makes it possible to look for a relationship between those two columns.

- If the points in a scatter plot appear to follow a straight line, it suggests that a linear relationship exists between those two columns. A number called a **correlation** can be used to summarize this relationship.
- r is the name of the **correlation statistic**. The r-value will always fall between −1 and +1. The sign tells us whether the correlation is positive or negative. Distance from 0 tells us the strength of the correlation.
  - −1 or +1 are the strongest possible negative and possible correlations.
  - 0 means no correlation.
- The correlation is **positive** if the point cloud slopes up as it goes farther to the right. This means larger y-values tend to go with larger x-values. It is **negative** if it slopes down as it goes farther to the right.
- If the points are tightly clustered around a line, it is a **strong** correlation. That means knowing the x-value gives us a pretty good idea of the y-value. If they are loosely scattered it is a **weak** correlation, and the y-value doesn't depend much on the x-value.
- Points that are far above or below the cloud of points in a scatter plot are called **outliers**.
- We graphically summarize this relationship by drawing a straight line through the data cloud, so that the vertical distance between the line and all the points taken together is as small as possible. This line is called the **line of best fit** and allows us to predict y-values based on x-values.

(Dis)Proving a Claim

"Smaller animals get adopted faster because they're cuter."

Do you agree? If so, why?

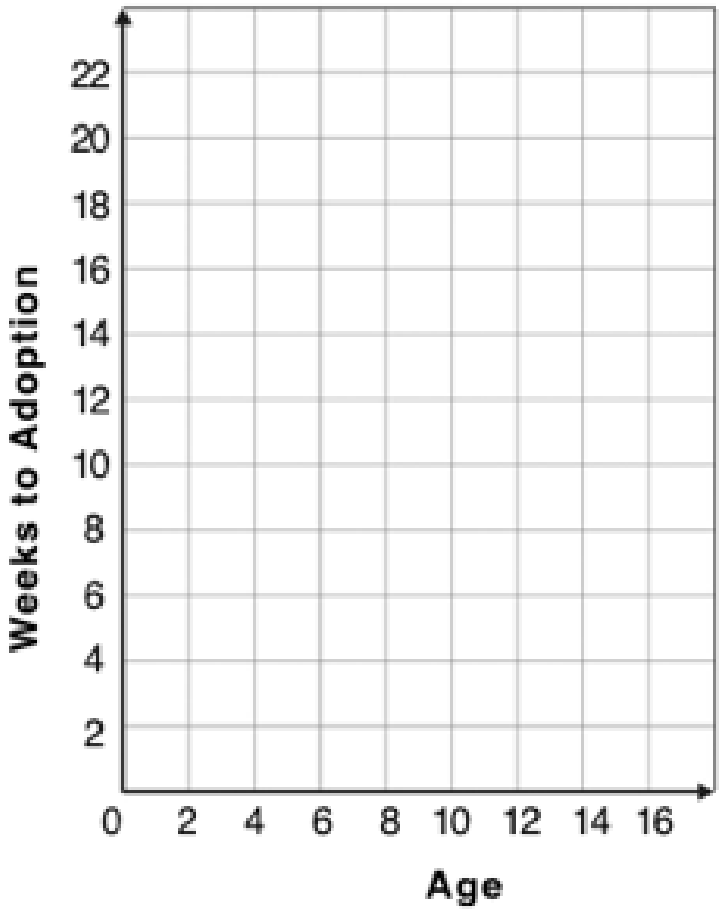
I hypothesize …

What would you look for in the dataset to see if you are right?

Creating a Scatter Plot

1. **For each row in the Sample Table on the left, add a point to the scatter plot on the right** . Use the values from the age column for the x-axis, and values from the weeks column for the y-axis.
2. Do you see a pattern? Do the points seem to go up or down as age increases to the right?
   - **Draw a cloud around all the points, and a line around which the cloud appears to be centered**
3. Does the line slope upwards or downwards? _____
4. Are the points tightly clustered around the line or loosely scattered? _____

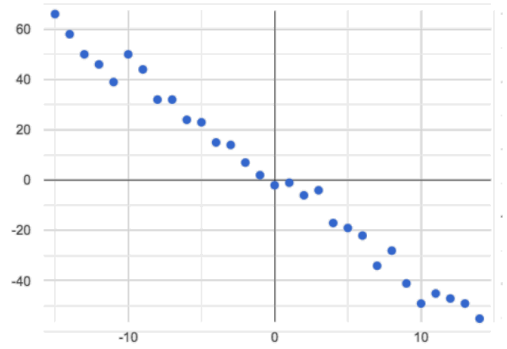
name	species	age	weeks
`"Sasha"`	`"cat"`	1	3
`"Boo-boo"`	`"dog"`	11	5
`"Felix"`	`"cat"`	16	4
`"Buddy"`	`"lizard"`	2	24
`"Nori"`	`"dog"`	6	9
`"Wade"`	`"cat"`	1	2
`"Nibblet"`	`"rabbit"`	6	12
`"Maple"`	`"dog"`	3	2



Identifying Form, Direction and Strength

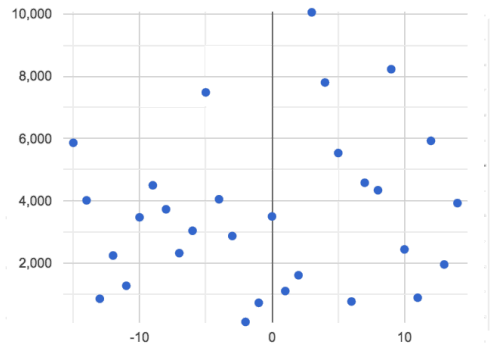
Can you identify the Form, Direction, & Strength of these displays? **Note:** If the form is non-linear, we shouldn't report direction - a curve may rise and then fall

A



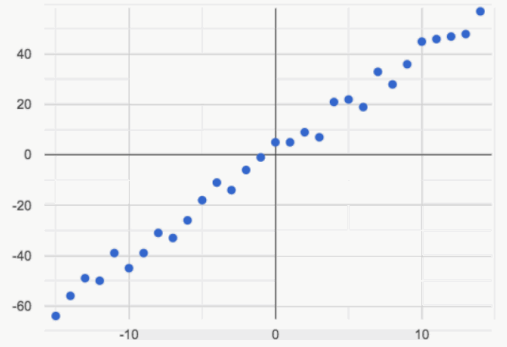
Form:	Linear	Non-Linear	None
Direction:	Positive	Negative	None
Strength:	Strong	Weak	None

B



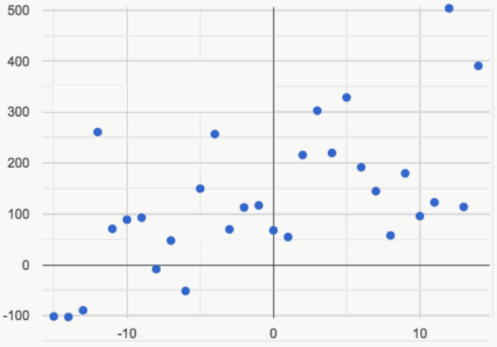
Form:	Linear	Non-Linear	None
Direction:	Positive	Negative	None
Strength:	Strong	Weak	None

C



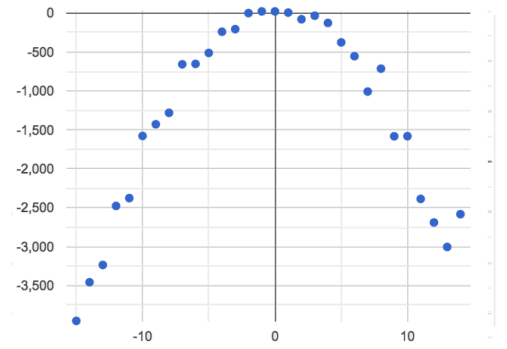
Form:	Linear	Non-Linear	None
Direction:	Positive	Negative	None
Strength:	Strong	Weak	None

D



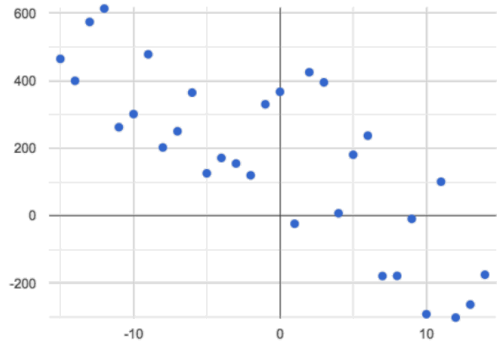
Form:	Linear	Non-Linear	None
Direction:	Positive	Negative	None
Strength:	Strong	Weak	None

E



Form:	Linear	Non-Linear	None
Direction:	Positive	Negative	None
Strength:	Strong	Weak	None

F



Form:	Linear	Non-Linear	None
Direction:	Positive	Negative	None
Strength:	Strong	Weak	None

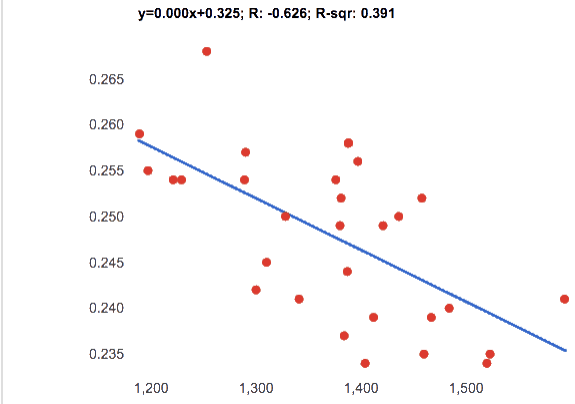
Identifying Form and r-Values

Can you identify the Form and r-Values of these displays?

If the form is linear, approximate the r-value to express Direction and Strength.

Reminder: An r-value close to -1 is a strong negative relationship, an r-value close to 0 is weak, and an r-value close to +1 is a strong positive! If the relationship's strength is moderate, the r-value will be closer to -0.5 or +0.5.

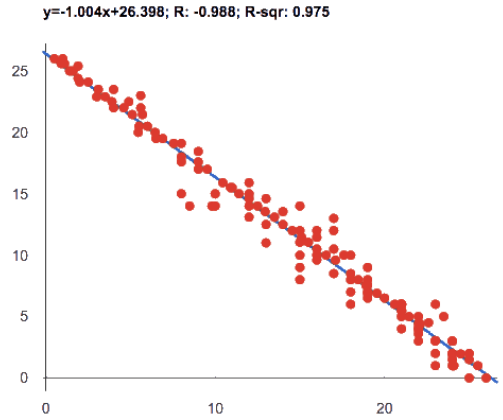
A



Form :

r close to :

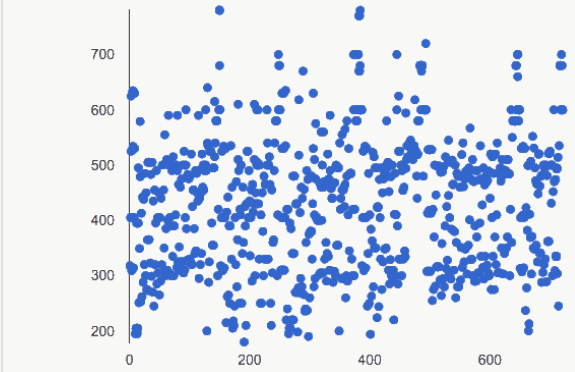
B



Form :

r close to :

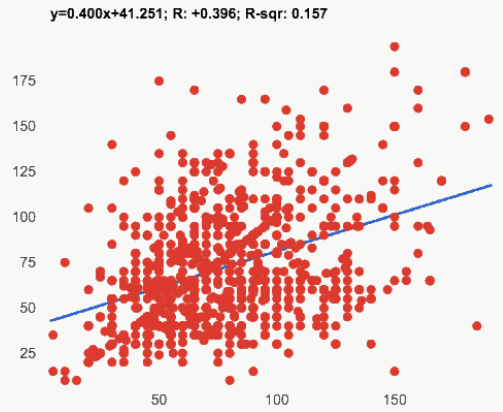
C



Form :

r close to :

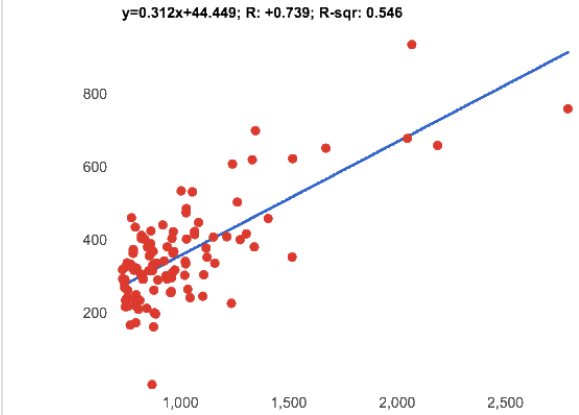
D



Form :

r close to :

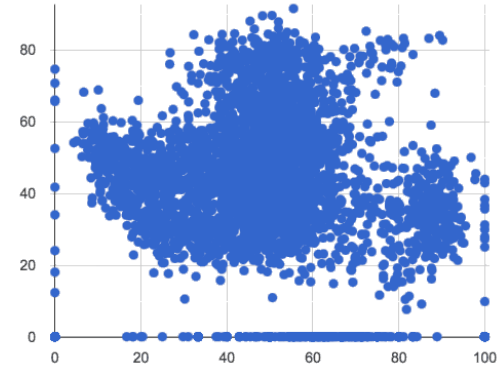
E



Form :

r close to :

F



Form :

r close to :

Correlations in My Dataset

1) There may be a correlation between _____ (column) and _____ (column).

I think it is a _____ (strong/weak), _____ (positive/negative) correlation,

because _____.

It might be stronger if I looked at _____ (a sample or extension of my data).

---

2) There may be a correlation between _____ (column) and _____ (column).

I think it is a _____ (strong/weak), _____ (positive/negative) correlation,

because _____.

It might be stronger if I looked at _____ (a sample or extension of my data).

---

3) There may be a correlation between _____ (column) and _____ (column).

I think it is a _____ (strong/weak), _____ (positive/negative) correlation,

because _____.

It might be stronger if I looked at _____ (a sample or extension of my data).

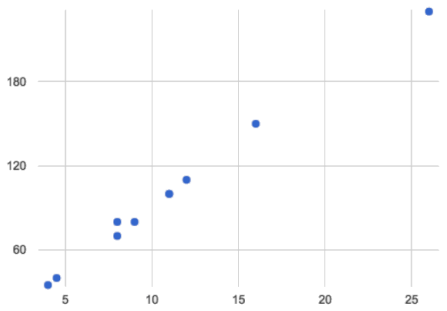
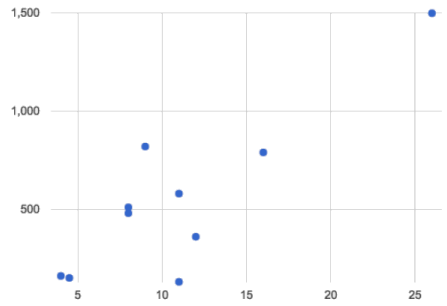
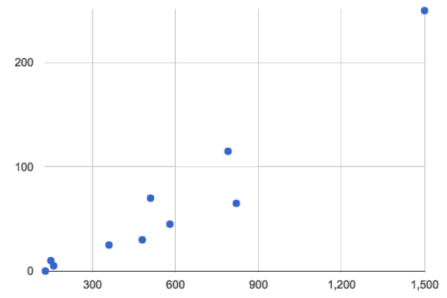
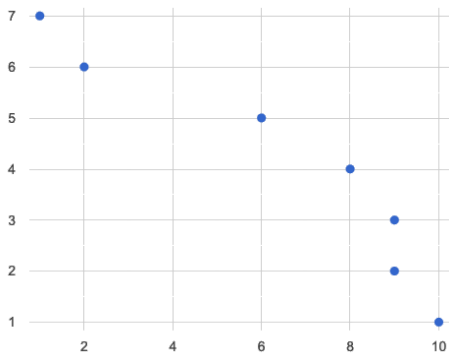
Computing Relationships

Linear Regression is a way of computing the **line of best fit**, which minimizes the *sum of the squares* of the vertical distances from the points to the line. Calculating the slope and intercept of this line is a task best left to computing or statistical software.

- **Slope** provides us with the easiest summary to grasp: it's how much we predict the y-variable (response variable) will increase or decrease for each unit that the x-variable (explanatory variable) increases.
- **Correlation is not causation!** Correlation only suggests that two column variables are related, but does not tell us if one causes the other. For example, hot days are correlated with people running their air conditioners, but air conditioners do not cause hot days!
- **Sample size matters!** The number of data values is also relevant. We'd be more convinced of a positive relationship in general between cat age and time to adoption if a correlation of +0.57 were based on 50 cats instead of 5.

Drawing Predictors

For each of the scatter plots below, draw a **predictor line** that seems like the best fit. Describe the correlation in terms of Direction and Strength, then estimate the *r*-value as being close to -1, -0.5, 0, +0.5, or +1.

A		**Direction**: Positive Negative None **Strength**: Strong Weak **r**: _____
B		**Direction**: Positive Negative None **Strength**: Strong Weak **r**: _____
C		**Direction**: Positive Negative None **Strength**: Strong Weak **r**: _____
D		**Direction**: Positive Negative None **Strength**: Strong Weak **r**: _____

Interpreting Regression Lines & r-Values

Each description on the left is written about the linear regression findings on the right. Fill in the blanks using the information in the line of best fit and the r-value.

1	For every additional Marvel Universe movie released each year, the average person is predicted to consume _____ [amount] _____ [more / fewer] pounds of sugar! This correlation is _____ [strong, moderate, weak, practically non-existent].	$f(x) = -3.19x + 12$ $r = -0.05$
2	Shoe size and height are _____ [strongly, moderately, weakly, not], _____ [positively / negatively] correlated. If person A is one size bigger than person B, we predict that they will be roughly _____ [amount] inches taller than person B as well.	$f(x) = 1.65x + 52$ $r = 0.89$
3	There is _____ [a strong, a moderate, a weak, no] relationship found between the number of Uber drivers in a city and the number of babies born each year.	$f(x) = -15.3x + 1150$ $r = 0.01$
4	The correlation between weeks-of-school-missed and SAT score is _____ [strong, moderate, weak, practically non-existent] and _____ [positive / negative]. For every week a student misses, we predict a more than a _____ [amount] point _____ [gain / drop] in their SAT score.	$f(x) = -5.35x - 16$ $r = -0.65$
5	There is a _____ [strong, moderate, weak, practically non-existent], _____ [positive / negative] correlation between the number of streaming video services someone has, and how much they weigh. For each service, we expect them to be roughly _____ [amount] pounds heavier.	$f(x) = 1.6x + 140$ $r = 0.12$

Regression Analysis in the Animals Dataset

1) I performed a linear regression on a sample of cats from the shelter [dataset or subset] and found a moderate (r=0.566), positive [weak/strong/moderate (R=…), positive/negative] correlation between age of the cats (in years) [x-axis] and number of weeks to adoption [y-axis].

I would predict that a 1 year [x-axis units] increase in age [x-axis] is associated with a 0.23 week [slope, y-units] increase [increase/decrease] in adoption time [y-axis].

---

2) I performed a linear regression on a sample of _____ [dataset or subset] and found a _____ [weak/strong/moderate (R=…), positive/negative] correlation between _____ [x-axis] and _____ [y-axis].

I would predict that a 1 _____ [x-axis units] increase in _____ [x-axis] is associated with a _____ [slope, y-units] _____ [increase/decrease] in _____ [y-axis].

---

3) I performed a linear regression on a sample of _____ [dataset or subset] and found a _____ [weak/strong/moderate (R=…), positive/negative] correlation between _____ [x-axis] and _____ [y-axis].

I would predict that a 1 _____ [x-axis units] increase in _____ [x-axis] is associated with a _____ [slope, y-units] _____ [increase/decrease] in _____ [y-axis].

Regression Analysis in Your Dataset

My Dataset is _____.

1) I performed a linear regression on _____ (dataset or subset) and found _____ (a weak/strong/moderate (R=…), positive/negative) correlation between _____ [x-axis] and _____ [y-axis].

I would predict that a 1 _____ [x-axis units] increase in _____ [x-axis] is associated with a _____ [slope, y-units] _____ [increase/decrease] in _____ [y-axis].

---

2) I performed a linear regression on _____ (dataset or subset) and found _____ (a weak/strong/moderate (R=…), positive/negative) correlation between _____ [x-axis] and _____ [y-axis].

I would predict that a 1 _____ [x-axis units] increase in _____ [x-axis] is associated with a _____ [slope, y-units] _____ [increase/decrease] in _____ [y-axis].

---

3) I performed a linear regression on _____ (dataset or subset) and found _____ (a weak/strong/moderate (R=…), positive/negative) correlation between _____ [x-axis] and _____ [y-axis].

I would predict that a 1 _____ [x-axis units] increase in _____ [x-axis] is associated with a _____ [slope, y-units] _____ [increase/decrease] in _____ [y-axis].

What's on your mind?

Contracts

Contracts tell us how to use a function. For example: `ellipse :: (Number, Number, String, String) -> Image` tells us that the name of the function is `ellipse`, it takes four inputs (two Numbers and two Strings), and it evaluates to an `Image`. From the contract, we know `ellipse(100, 50, "outline", "red")` will evaluate to an `Image`.

Name	Domain	Range
`# string-equal`	`::`	`->`
`#`		
`# string-contains`	`::`	`->`
`#`		
`# num-sqr`	`::`	`->`
`#`		
`# num-sqrt`	`::`	`->`
`#`		
`# string-length`	`::`	`->`
`#`		
`# triangle`	`::`	`->`
`#`		
`# star`	`::`	`->`
`#`		
`# circle`	`::`	`->`
`#`		
`# square`	`::`	`->`
`#`		

Contracts

Contracts tell us how to use a function. For example: `ellipse :: (Number, Number, String, String) -> Image` tells us that the name of the function is `ellipse`, it takes four inputs (two Numbers and two Strings), and it evaluates to an `Image`. From the contract, we know `ellipse(50, 100, "solid", "teal")` will evaluate to an `Image`.

Name		Domain		Range
`# rectangle`	`::`		`->`	
`#`				
`# rhombus`	`::`		`->`	
`#`				
`# ellipse`	`::`		`->`	
`#`				
`# text`	`::`		`->`	
`#`				
`# regular-polygon`	`::`		`->`	
`#`				
`# radial-star`	`::`	Number, Number, Number, String, String	`->`	
`#`				
`# image-url`	`::`		`->`	
`#`				
`# scale`	`::`		`->`	
`#`				
`# rotate`	`::`		`->`	
`#`				

Contracts

Contracts tell us how to use a function. For example: `ellipse :: (Number, Number, String, String) -> Image` tells us that the name of the function is `ellipse`, it takes four inputs (two Numbers and two Strings), and it evaluates to an `Image`. From the contract, we know `ellipse(100, 50, "solid", "fuchsia")` will evaluate to an `Image`.

Name	Domain	Range
`# overlay`	`::`	`->`
`#`		
`# put-image`	`::`	`->`
`#`		
`# flip-horizontal`	`::`	`->`
`#`		
`# flip-vertical`	`::`	`->`
`#`		
`# above`	`::`	`->`
`#`		
`# beside`	`::`	`->`
`#`		
`#`	`::`	`->`
`#`		
`#`	`::`	`->`
`#`		
`#`	`::`	`->`
`#`		

Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
`num-min`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`num-max`	`::`	`(a :: Number, b:: Number)`	`->`	`Number`
`count`	`::`	`(t :: Table, col :: String)`	`->`	`Table`
`mean`	`::`	`(t :: Table, col :: String)`	`->`	`Number`
`median`	`::`	`(t :: Table, col :: String)`	`->`	`Number`
`modes`	`::`	`(t :: Table, col :: String)`	`->`	`List<Number>`
`bar-chart`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`pie-chart`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`histogram`	`::`	`(t :: Table, values :: String, bin-width :: Number)`	`->`	`Image`

Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
`box-plot`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`modified-box-plot`	`::`	`(t :: Table, col :: String)`	`->`	`Image`
`scatter-plot`	`::`	`(t :: Table, labels :: String, xs :: String, ys :: String)`	`->`	`Image`
`image-scatter-plot`	`::`	`(t :: Table, xs :: String, ys :: String, f :: (Row -> Image))`	`->`	`Image`
`r-value`	`::`	`(t :: Table, xs :: String, ys :: String)`	`->`	`Number`
`lr-plot`	`::`	`(t :: Table, labels :: String, xs :: String, ys :: String)`	`->`	`Image`
`random-rows`	`::`	`(t :: Table, num-rows :: Number)`	`->`	`Table`
`<Table>.row-n`	`::`	`(n :: Number)`	`->`	`Row`
`<Table>.order-by`	`::`	`(col :: String, increasing :: Boolean)`	`->`	`Table`

Contracts

Contracts tell us how to use a function. For example: `num-min :: (a :: Number, b :: Number) -> Number` tells us that the name of the function is `num-min`, it takes two inputs (both Numbers), and it evaluates to a `Number`. From the contract, we know `num-min(4, 6)` will evaluate to a `Number`. **Use the blank line under each contract for notes or sample code for that function!**

Name		Domain		Range
`<Table>.filter`	`::`	`(test :: (Row -> Boolean))`	`->`	`Table`
`<Table>.build-column`	`::`	`(col :: String, builder :: (Row -> Any))`	`->`	`Table`
`bar-chart-summarized`	`::`	`(t :: Table, labels :: String, values :: String)`	`->`	`Image`
`pie-chart-summarized`	`::`	`(t :: Table, labels :: String, values :: String)`	`->`	`Image`

These materials were developed partly through support of the National Science Foundation, (awards 1042210, 1535276, 1648684, and 1738598),