Name: ______________________________

BOOTSTRAP
DATA SCIENCE

Student Workbook

Spring, 2022 - Pyret Edition

BOOTSTRAP

Equity • Scale • Rigor

Workbook v1.5

Brought to you by the Bootstrap team:

- Emmanuel Schanzer
- Kathi Fisler
- Shriram Krishnamurthi
- Dorai Sitaram
- Joe Politz
- Ben Lerner
- Nancy Pfenning
- Flannery Denny

Visual Designer: Colleen Murphy

Intro to CODAP & Displaying Categorical Data

Intro to CODAP

- With a table open in CODAP, select the "graph" icon to produce a scatterplot of *randomly distributed* data points.
- Drag attributes/columns to the axes (or select from a drop-down menu of attributes/columns by clicking the axes) to organize the data so that it is no longer randomly distributed.
- Once the data is organized, manipulate it further by selecting the graph menu icons:
    - **the ruler icon** provides options for calculating statistics such as mean, median, and standard deviation
    - for *some* datasets (those with strong correlations), the ruler icon will offer *additional* statistical computations (such as a least squares line or regression line)
    - the **bar graph icon** allows new configurations of the data. For instance, select this option to group data points into bins or create a bar for each point. Clicking on the bar graph icon a second time (for instance, after data is grouped into bins) allows the creation of a histogram (by fusing the dots into bars).

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** , we often rely on **bar charts** and **pie charts** . CODAP, however, only has the capacity to create **bar charts** .

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).

Exploring Displays

Using the Animals Dataset in CODAP, make each type of display below. Then sketch the displays and answer the questions.

Bar Charts	Box Plots
Sketch a bar chart here.	Sketch a box plot here.
Displays ___ column(s) of ________________ data. What does this display tell us? ____ ____ ____ ____	Displays ___ column(s) of ________________ data. What do you think this display tells us? ____ ____ ____ ____
Histogram	**Scatterplot**
Sketch a histogram here	Sketch a scatterplot 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.

Least Square Line
Sketch a least square line here.
Displays ____ column(s) of ________________ data. What do you think this display tells us? ____ ____ ____ ____

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
  ```

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: is-dog / is-female

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: is-old / name-has-s

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 dataset. 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 population 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` , `is-fixed` , and `name-has-s` . 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?

Design Recipe

Directions :

Transformer (check one) □

□ Filter

□ Transform

□ Build

function name

Example Tables □

What gets filtered/transformed/built? In the sample tables below, add the relevant columns.

Formula Expression □

domain -> *range*

# *what does the function do?*

Directions :

Transformer (check one) □

□ Filter

□ Transform

□ Build

function name

Example Tables □

What gets filtered/transformed/built? In the sample tables below, add the relevant columns.

Formula Expression □

domain -> *range*

# *what does the function do?*

Design Recipe

Directions :

Transformer (check one) □

□ Filter

□ Transform

□ Build

function name

Example Tables □

What gets filtered/transformed/built? In the sample tables below, add the relevant columns.

Formula Expression □

->

domain _range_

#

what does the function do?

Directions :

Transformer (check one) □

□ Filter

□ Transform

□ Build

function name

Example Tables □

What gets filtered/transformed/built? In the sample tables below, add the relevant columns.

Formula Expression □

->

domain _range_

#

what does the function do?

Design Recipe

Directions :

Transformer (check one) □

□ Filter

□ Transform

□ Build

function name

Example Tables □

What gets filtered/transformed/built? In the sample tables below, add the relevant columns.

Formula Expression □

->

domain *range*

#

what does the function do?

Directions :

Transformer (check one) □

□ Filter

□ Transform

□ Build

function name

Example Tables □

What gets filtered/transformed/built? In the sample tables below, add the relevant columns.

Formula Expression □

->

domain *range*

#

what does the function do?