

Misc Updates

- Updated office hours
- Don't forget lab attendance is required!!
 - Please email your TAs in advance if you'll be late or can't make it!
- HW 1 out today, due next week Wednesday
 - Known issue with the autograder for Q4.2, it's fine
 - Remember to run the last cell in the notebook

Office Hours:

- M 2:30pm-4:00pm, Milstein Center 503 (Erin Ma)
- Tu 2:00pm-3:30pm, Milstein Center 503 (Amaya Kejriwal)
- W 3:00pm-4:00pm, Milstein Center 512 (Prof. Eysa Lee)
- Th 10:00am-11:30am, Virtual (via Google Meet) (Ken Mah)
- Th 1:00pm-2:30pm, Milstein Center 503 (Justin Zeng)
- MW 2:45pm-3:45pm, Milstein Center 511 (Prof. Murad Megjhani)

Last Class Recap

- Built-in Data Types
 - Numbers (integers and floats)
 - Strings
 - Booleans
- Programming basics
 - Functions
 - Assignments

Let's say you have defined the following variables in your notebook

$$x = 3$$
 $y = 4'$
 $z = 5.6'$

What would the source of the error in these examples?

Let's say you have defined the following variables in your notebook

$$x = 3$$
 $y = 4'$
 $z = 5.6'$

Take 3 min to work on your own or with a neighbor!

What would the source of the error in this example:

$$x + \lambda$$

Let's say you have defined the following variables in your notebook

$$x = 3$$
 $y = 4'$
 $z = 5.6'$

Take 3 min to work on your own or with a neighbor!

What would the source of the error in this example:

$$x + int(y + z)$$

Let's say you have defined the following variables in your notebook

$$x = 3$$
 $y = 4'$
 $z = 5.6'$

Take 3 min to work on your own or with a neighbor!

What would the source of the error in this example:

$$str(x) + int(y)$$

Let's say you have defined the following variables in your notebook

$$x = 3$$
 $y = 4'$
 $z = 5.6'$

Take 3 min to work on your own or with a neighbor!

What would the source of the error in this example:

$$y + float(z)$$

Importing Libraries

- Recall we can import libraries
- Can also do import library as shorter name
 - e.g., import numpy as np

- import numpy
 numpy.sqrt(4)
 2.0
- Can also specific functions to import from a specific library
 - e.g., from math import sqrt
 - Use * to import all functions
 - e.g., from datascience import *
 - Be careful doing this with too many libraries at once!

Arrays

Arrays

- Arrays are a sequence of values
 - e.g., ["Mystery", "Abby", "Jinu", "Baby", "Romance"] or [1,2,3,5]
 - Elements of an array should have the same type
- We'll mostly use NumPy arrays
 - Can make arrays using datascience.make_array or numpy.array
- Can perform component-wise arithmetic
 - Note this only works for numpy arrays but not built-in Python lists!

```
from datascience import *
onetwothree = make_array(1,2,3)
onetwothree * 2
array([2, 4, 6])
```

```
from datascience import *
  onetwothree = make_array(1,2,3)
  twothreefour = make_array(2,3,4)
  onetwothree + twothreefour

array([3, 5, 7])
```

```
onetwothree = [1,2,3]
twothreefour = [2,3,4]
onetwothree + twothreefour
[1, 2, 3, 2, 3, 4]
```

Arrays

- Many useful functions for operating on arrays
 - Helpful to be aware of, but you do not need to memorize them!

Function	Description
np.prod	Multiply all elements together
np.sum	Add all elements together
np.all	Test whether all elements are true values (non-zero numbers are true)
np.any	Test whether any elements are true values (non-zero numbers are true)
np.count_nonzero	Count the number of non-zero elements

Function	Description
np.char.lower	Lowercase each element
np.char.upper	Uppercase each element
np.char.strip	Remove spaces at the beginning or end of each element
np.char.isalpha	Whether each element is only letters (no numbers or symbols)
np.char.isnumeric	Whether each element is only numeric (no letters)

Function	Description
np.diff	Difference between adjacent elements
np.round	Round each number to the nearest integer (whole number)
np.cumprod	A cumulative product: for each element, multiply all elements so far
np.cumsum	A cumulative sum: for each element, add all elements so far
np.exp	Exponentiate each element
np.log	Take the natural logarithm of each element
np.sqrt	Take the square root of each element
np.sort	Sort the elements

Function	Description
np.char.count	Count the number of times a search string appears among the elements of an array
np.char.find	The position within each element that a search string is found first
np.char.rfind	The position within each element that a search string is found last
np.char.startswith	Whether each element starts with the search string

Ranges

A range is an array of consecutive numbers

```
- np.arange (end)
```

Create an array of increasing integers from 0 up to end

```
-np.arange(start, end)
```

Create an array of increasing integers from start up to end

```
-np.arange(start, end, step)
```

A range where step is added between consecutive values

The range always includes start but excludes end

Tables

Tables

A table is a way of representing data sets

- Each row is an individual
- Each column is an attribute of the individual

Name	Age	Coloring	Favorite Food
Gertrude	15 yrs	Tuxedo	Milk
Ruby	14 yrs	Tuxedo	Potato chips
Corina	6 yrs	Dilute Tortoiseshell	Kibble
Frito	1 yr	Tabby	Cheese

Creating datascience Tables

Create an empty table using Table ()

Each column of a table is an array and with_columns creates a table with the array of values as a new column

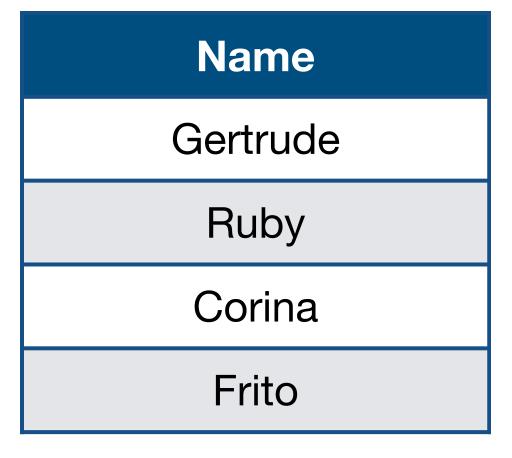
Name	Description	Input	Output
Table()	Create an empty table, usually to extend with data (Ch 6)	None	An empty Table
Table().read_table(filename)	Create a table from a data file (Ch 6)	string: the name of the file	Table with the contents of the data file
<pre>tbl.with_columns(name, values) tbl.with_columns(n1, v1, n2, v2,)</pre>	A table with an additional or replaced column or columns. name is a string for the name of a column, values is an array (Ch 6)	 string: the name of the new column; array: the values in that column 	Table : a copy of the original Table with the new columns added

Creating datascience Tables

Create an empty table using Table ()

Each column of a table is an array and with_columns creates a table with the array of values as a new column

```
Table().with_columns("Name", make_array("Gertrude",
"Ruby", "Corina", "Frito"))
```



Creating datascience Tables

Create an empty table using Table ()

Each column of a table is an array and with_columns creates a table with the array of values as a new column

```
Table().with_columns("Name", make_array("Gertrude",
"Ruby", "Corina", "Frito"),
"Age", make array(15,14,6,1))
```

Name	Age
Gertrude	15
Ruby	14
Corina	6
Frito	1

More Ways to Create Tables

- Read from a CSV file

```
- Table.read_table(filename)
```

- Create a new table from an existing table. Let tbl be a table and c, c1, c2 be column names or indices

Table with only columns c1, c2, ...

```
- tbl.select(c1 ,c2, ...)
```

```
- tbl.drop(c1, c2, ...) Table without columns c1, c2,...
```

```
- tbl.where(c, predicate)
```

Only rows in the table where the value in column c satisfies the predicate

only the specified rows

https://www.data8.org/sp22/python-reference.html

Filtering

Table.where Predicates

Any of these predicates can be negated by adding not_ in front of them, e.g. are.not_equal_to(Z) or are.not_containing(S) .

Predicate	Description
are.equal_to(Z)	Equal to Z
are.not_equal_to(Z)	Not equal to Z
are.above(x)	Greater than x
are.above_or_equal_to(x)	Greater than or equal to x
are.below(x)	Less than x
are.below_or_equal_to(x)	Less than or equal to x
are.between(x,y)	Greater than or equal to x and less than y
are.between_or_equal_to(x,y)	Greater than or equal to $\overline{\mathbf{x}}$, and less than or equal to $\overline{\mathbf{y}}$
are.contained_in(A)	Is a substring of A (if A is a string) or an element of A (if A is a list/array)
are.containing(S)	Contains the string S
<pre>are.strictly_between(x,y)</pre>	Greater than x and less than y

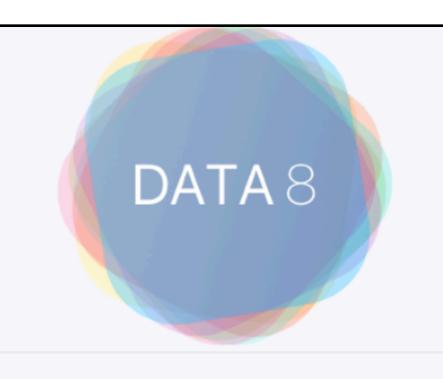
Table Methods

Recall each column in a Table is an array

- column takes a label or index and returns an array
- Array methods work on data in the columns
 - e.g., sum, min, max, average

Python Reference

https://www.data8.org/fa25/reference/



- A Home
- Weekly Calendar & OH
- Syllabus
- Staff
- **Resources**
- FAQs & Debugging
- 2 Python Reference
- Textbook

Q Search Data 8

Detailed Python Reference Sheet

Created by Nishant Kheterpal and Jessica Hu; contributions by Jonathan Ferrari, Edwin Vargas and Bing Concepcion Updated and maintained by Marissa Lumpkin and Isaac Chung

TABLE OF CONTENTS

- Detailed Python Reference Sheet
 - a Abbreviated Reference Sheet
 - b Table Functions and Methods
 - c String Methods
 - d Array Functions and Methods
 - e Table Filtering Predicates
 - f Miscellaneous Functions
 - g JupyterHub Keyboard Shortcuts

Class Activity

Next Class

- Last Wednesday
 - Jupyter Notebooks
 - Expressions
 - Data Types
- Today
 - Tables (and arrays)
- Wednesday
 - Charts & Visualization