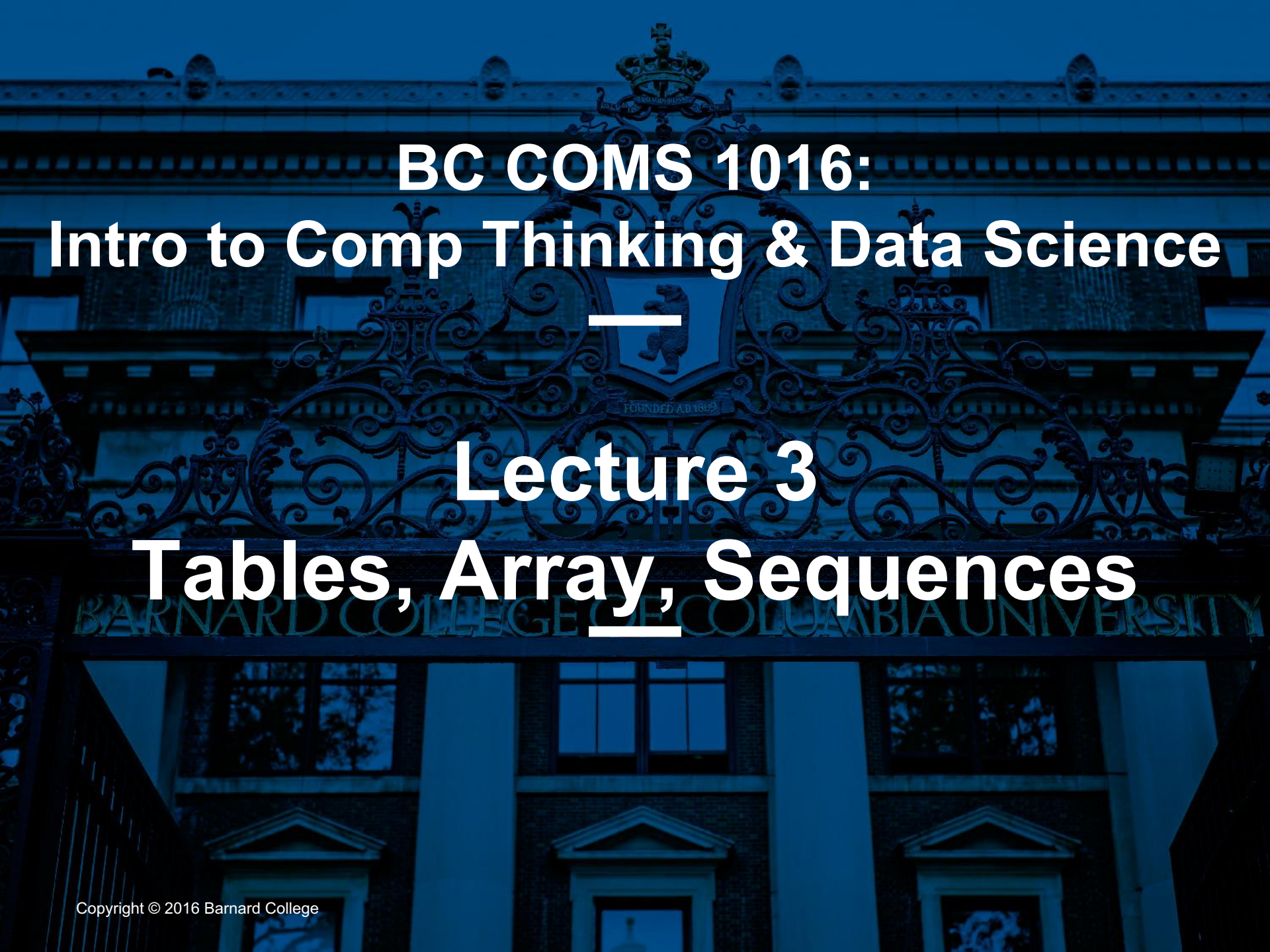


BC COMS 1016:
Intro to Comp Thinking & Data Science

—

Lecture 3
Tables, Array, Sequences

Copyright © 2016 Barnard College

Reminders

- HW00 due Thursday night
  - Individual assignment
  - Ask questions in EdStem #homeworks
  - Answer your peers questions as well
    - Can count for participation grade
- Lab00 due last night

Copyright © 2016 Barnard College

Gradescope Upload

- Unzip the file from JupyterHub
- Upload the pdf and .ipynb files to gradescope
- Don’t include a “.OTTER_log” file

Copyright © 2016 Barnard College – Adam Poliak

Office Hours

- Adam: **Tuesdays, 1pm - 2pm**
- Dingwen: **Wednesdays, 10am - 11:30 am**
- Saloni: **Tuesdays, 4pm - 5.30pm**
- Yosha: **Fridays, 4pm - 5.30pm**
- Pranathi Srirangam: **Mondays, 8.30pm - 9.30pm**
- Esha Julka: **Wednesdays, 4.30pm - 5.30pm**
- Xueqing Ma: **Mondays, 4pm - 5pm**

Barnard CS Help Room

- Mondays: 1 pm - 3 pm
- Tuesdays: 12pm - 2pm, 5pm - 7pm
- Wednesdays: 2pm - 5pm
- Thursdays: 10 am - 12 pm, 4pm - 6pm

Autograders/Grading

Question 1.1. In the next cell, assig

1. the **absolute value** of $2^5 - 2^1$
2. $5 \times 13 \times 31 + 5$.

Try to use just one statement (one l

```
new_year = ...
new_year
```

```
grader.check("q1_1")
```

Copyright © 2016 Barnard College

Autograders/Grading – Error 1

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-1ad9a283f073> in <module>()
----> 1 grader.check("q1_1")

NameError: name 'grader' is not defined
```

```
# Initialize Otter
import otter
grader = otter.Notebook()
```

Copyright © 2016 Barnard College

Autograders/Grading – Error 2

```
NameError: name 'new_year' is not defined
```

```
In [ ]: new_year = ...
        new_year

In [2]: grader.check("q1_1")
```

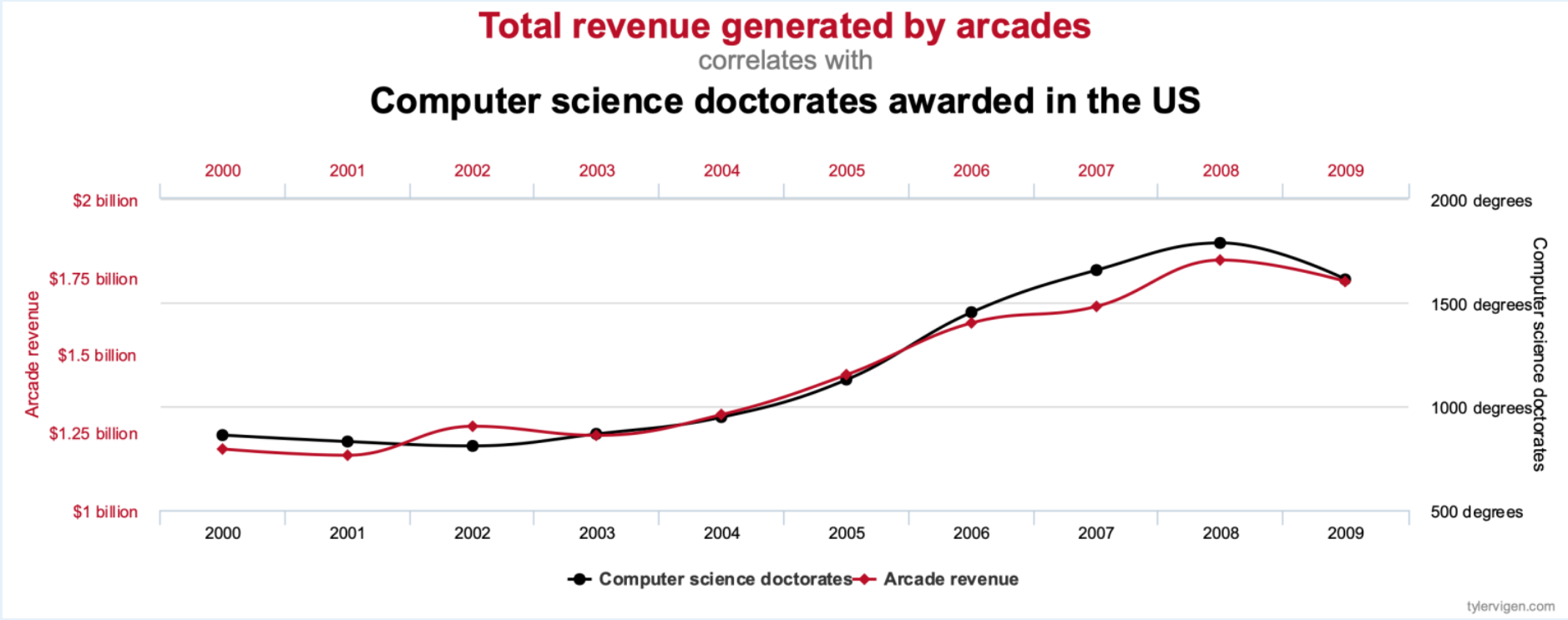
Copyright © 2016 Barnard College

Grading based on autograder

- Before we “publish” scores
  - Visible:
    - Status of tests (pass/fail)
    - Errors of failing test
  - Not visible
    - points associated with the tests
- Publish results after the assignment submission is closed
  - At least 2 days after deadline
    - Likely more

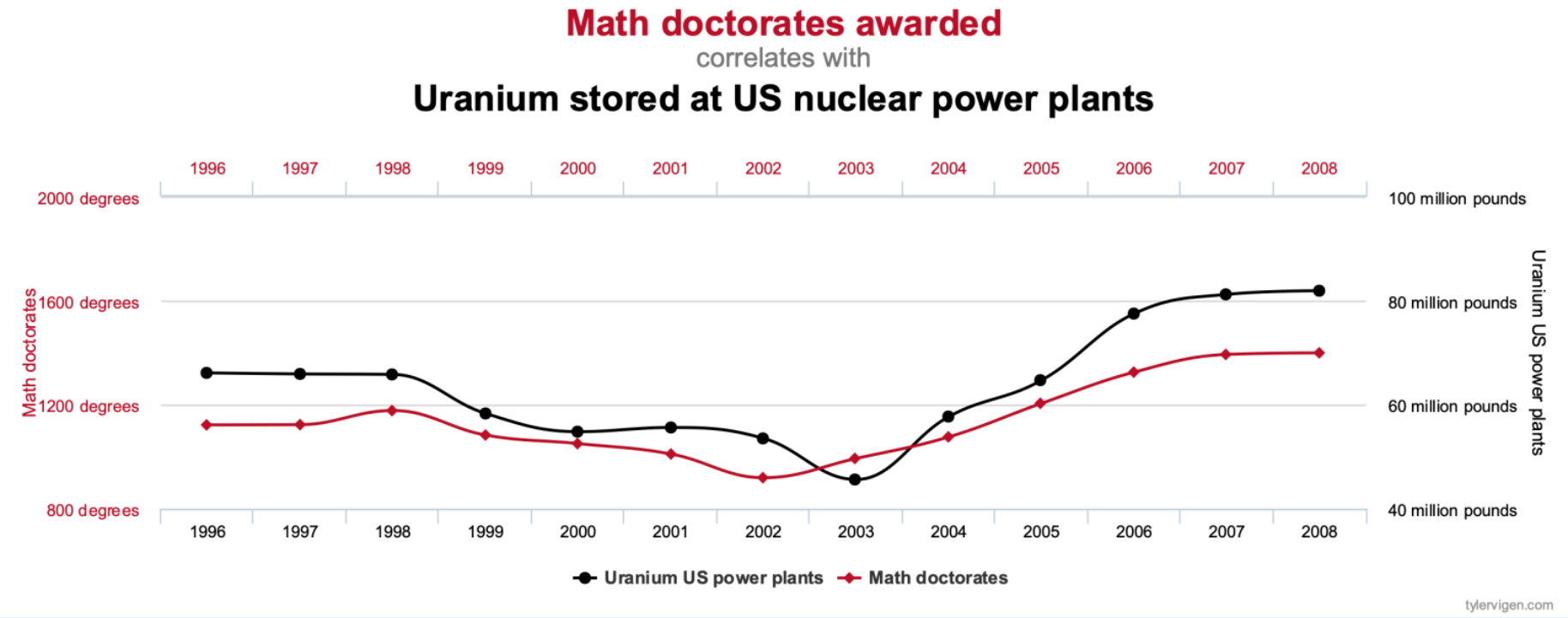
Copyright © 2016 Barnard College

Cause & Effect



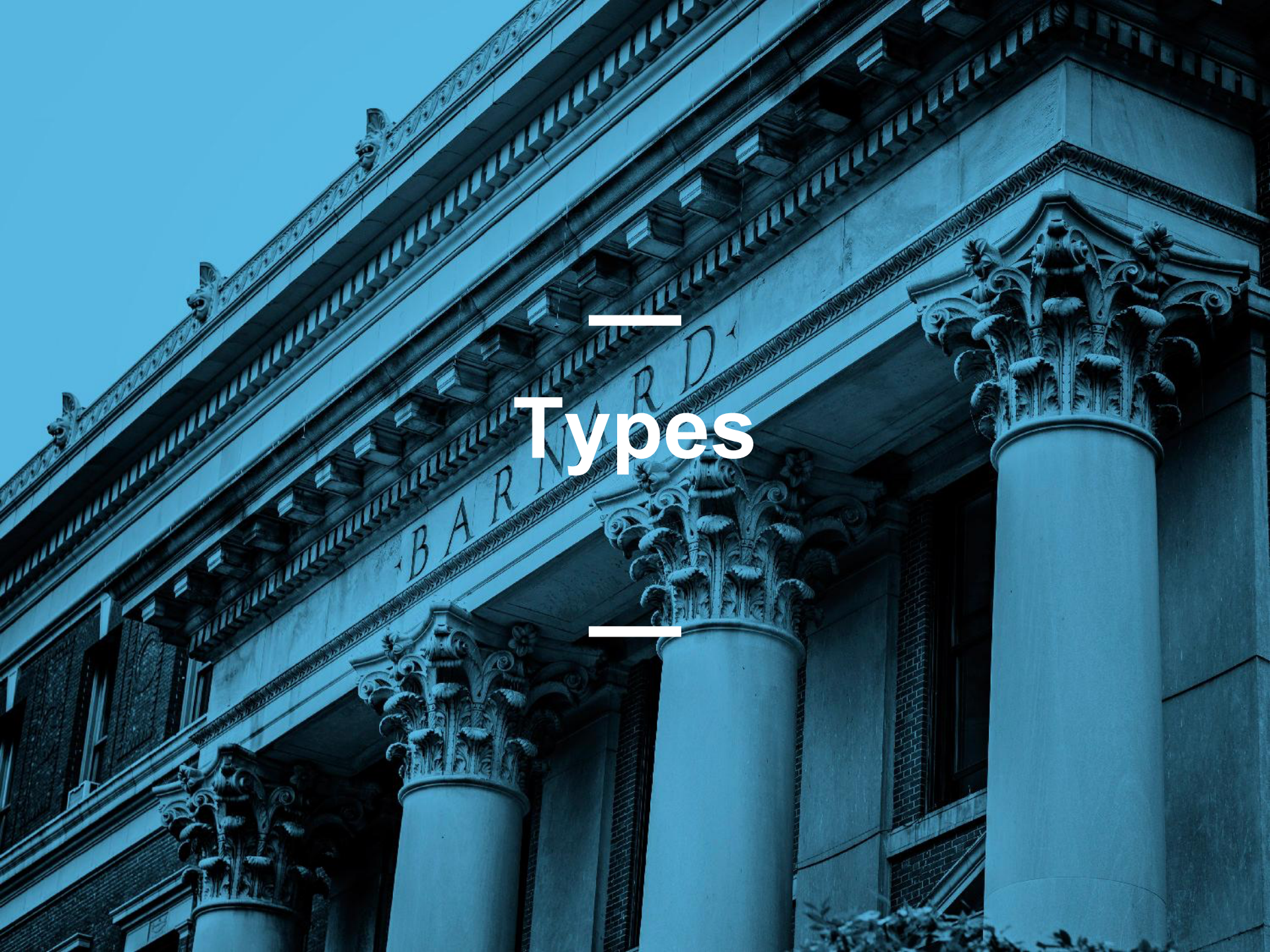
Copyright © 2016 Barnard College

Cause & Effect



https://www.tylervigen.com/spurious-correlations

Copyright © 2016 Barnard College



Types

Types – Every value has a type

We've seen 5 types so far:

- int: 2
- float: 2.2
- str: 'Red fish, blue fish'
- builtin_function_or_method: abs, max, min

Copyright © 2016 Barnard College

Types – Every value has a type

The type function tells you the type of a value

- type(2)
- type(2+2)

An expression's "type" is based on its value

- x = 2
- type(x) = ???

Copyright © 2016 Barnard College

Conversions

Strings that contain numbers can be converted to numbers

- int("12")
- float("1.2")
- ~~float("one point two")~~ # Not a good idea

Copyright © 2016 Barnard College

Conversions

Any value can be converted to a string

- str(6)

Numbers can be converted to other numeric types

- float(1)
- int(2.3). # DANGER: why is this a bad idea

Copyright © 2016 Barnard College

Copyright © 2016 Barnard College

Table Structure

- A Table is a sequence of labeled columns
- Row: represents one individual
- Column: represents one attribute of the individuals

Name	Code	Area (m2)
California	CA	163696
Nevada	NV	110567

Copyright © 2016 Barnard College

Creating a Table

- Table.read_table(filename) – reads a table from a spreadsheet

- Table() – an empty table

Copyright © 2016 Barnard College

Table methods

- Creating and extending tables:
  - Table().with_column and Table.read_table
- Finding the size:
  - num_rows , num_columns
- Referring to columns: labels, relabeling and indices
  - labels and relabeled; column indices start at 0

Copyright © 2016 Barnard College

Some Table Operations

- t.select(label) – constructs a new table with just the specified columns
- t.drop(label) – constructs a new table in which the specified columns are omitted
- t.sort(label) – constructs a new table with rows sorted by the specified column
- t.where(label, condiction) – constructs a new table with just the rows that match the condition

- These operations create a new table

Copyright © 2016 Barnard College

Array

Copyright © 2016 Barnard College

Arrays

An array contains a sequence of values

- All elements of an array should have the same type
- Arithmetic is applied to each element individually
- Adding arrays add elements (**if same length!)**
- A column of a table is in an array

Copyright © 2016 Barnard College

Ranges

A range is an array of consecutive numbers

- np.arange(end):
  An array of increasing integers from 0 up to end
- np.arange(start, end):
  An array of increasing integers from start up to end
- np.arrange(start, end, step):
  A range with step between consecutive values

The range always include start but excludes end

Copyright © 2016 Barnard College

Array Functions & Methods

Name	Chapter	Description
`max(array)`	3.3	Returns the maximum value of an array
`min(array)`	3.3	Returns the minimum value of an array
`sum(array)`	3.3	Returns the sum of the values in an array
`abs(num), np.abs(array)`	3.3	Take the absolute value of number or each number in an array.
`round(num), np.round(array)`	3.3	Round number or array of numbers to the nearest integer.
`len(array)`	3.3	Returns the length (number of elements) of an array
`make_array(val1, val2, ...)`	5	Makes a numpy array with the values passed in
`np.average(array) np.mean(array)`	5.1	Returns the mean value of an array
`np.std(array)`	14.2	Returns the standard deviation of an array
`np.diff(array)`	5.1	Returns a new array of size `len(arr)-1` with elements equal to the difference between adjacent elements; val_2 - val_1, val_3 - val_2, etc.
`np.sqrt(array)`	5.1	Returns an array with the square root of each element
`np.arange(start, stop, step) np.arange(start, stop) np.arange(stop)`	5.2	An array of numbers starting with `start`, going up in increments of `step`, and going up to but excluding `stop`. When `start` and/or `step` are left out, default values are used in their place. Default step is 1; default start is 0.
`array.item(index)`	5.3	Returns the i-th item in an array (remember Python indices start at 0!)
`np.random.choice(array, n) np.random.choice(array)`	9	Picks one (by default) or some number 'n' of items from an array at random. By default, with replacement.
`np.count_nonzero(array)`	9	Returns the number of non-zero (or `True`) elements in an array.
`np.append(array, item)`	9.2	Returns a copy of the input array with `item` (must be the same type as the other entries in the array) appended to the end.
`percentile(percentile, array)`	13.1	Returns the corresponding percentile of an array.

Copyright © 2016 Barnard College

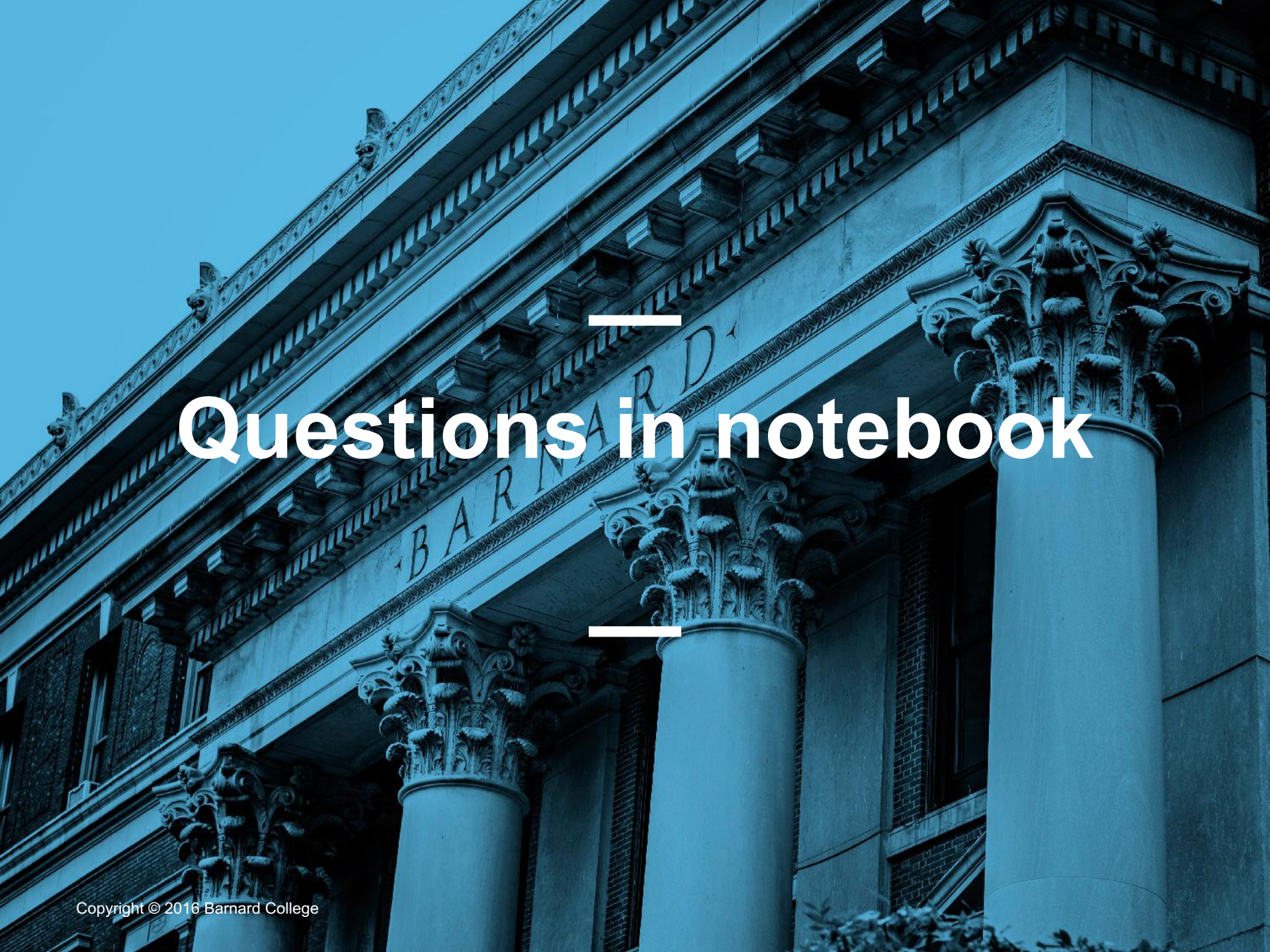
Tables & Arrays

Copyright © 2016 Barnard College

Table methods

- Accessing data in a column
  - Column takes a label or index and returns an array
- Using array methods to work with data in columns
  - item, sum, min, max, and so on
- Creating new tables containing some of the original columns
  - select, drop

Copyright © 2016 Barnard College



Questions in notebook

Copyright © 2016 Barnard College

Questions:

The table nba has columns

PLAYER, POSITION, and SALARY

```
table = Table.read_table('https://www.inferentialthinking.com/data/nba_salaries.csv')
```

1. Create an array containing the names of all centers (C) who make more than $15M/year

```
centers = table.where('POSITION', 'C')
centers.where('\'15-\'16 SALARY', are.above(15)).column('PLAYER')
```

Answer:

'Dwight Howard', 'Roy Hibbert', 'Marc Gasol', 'Enes Kanter', 'DeMarcus Cousins'

Copyright © 2016 Barnard College