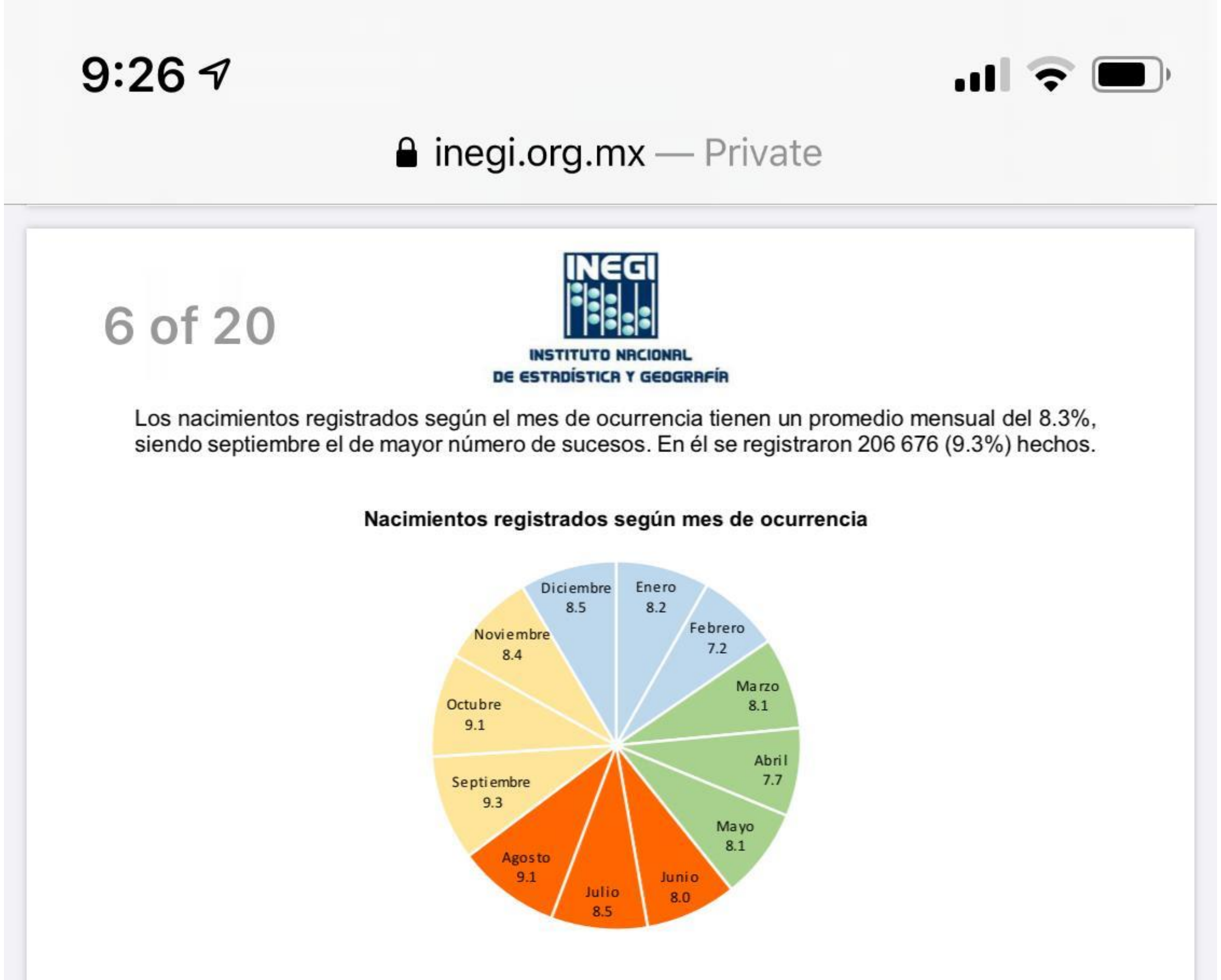
# Lecture 2

# Causality

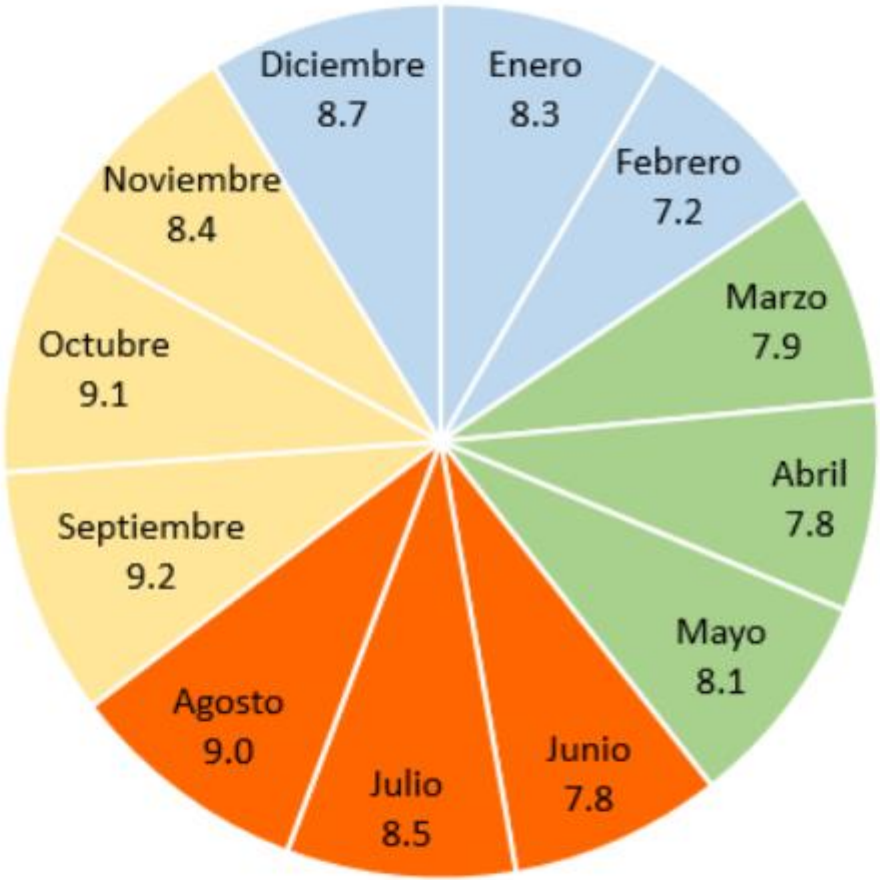


INEGI
INSTITUTO NACIONAL
DE ESTADÍSTICA Y GEOGRAFÍA

Los nacimientos registrados según el mes de ocurrencia tienen un promedio mensual del 8.3%, siendo septiembre el de mayor número de sucesos. En él se registraron 206 676 (9.3%) hechos.

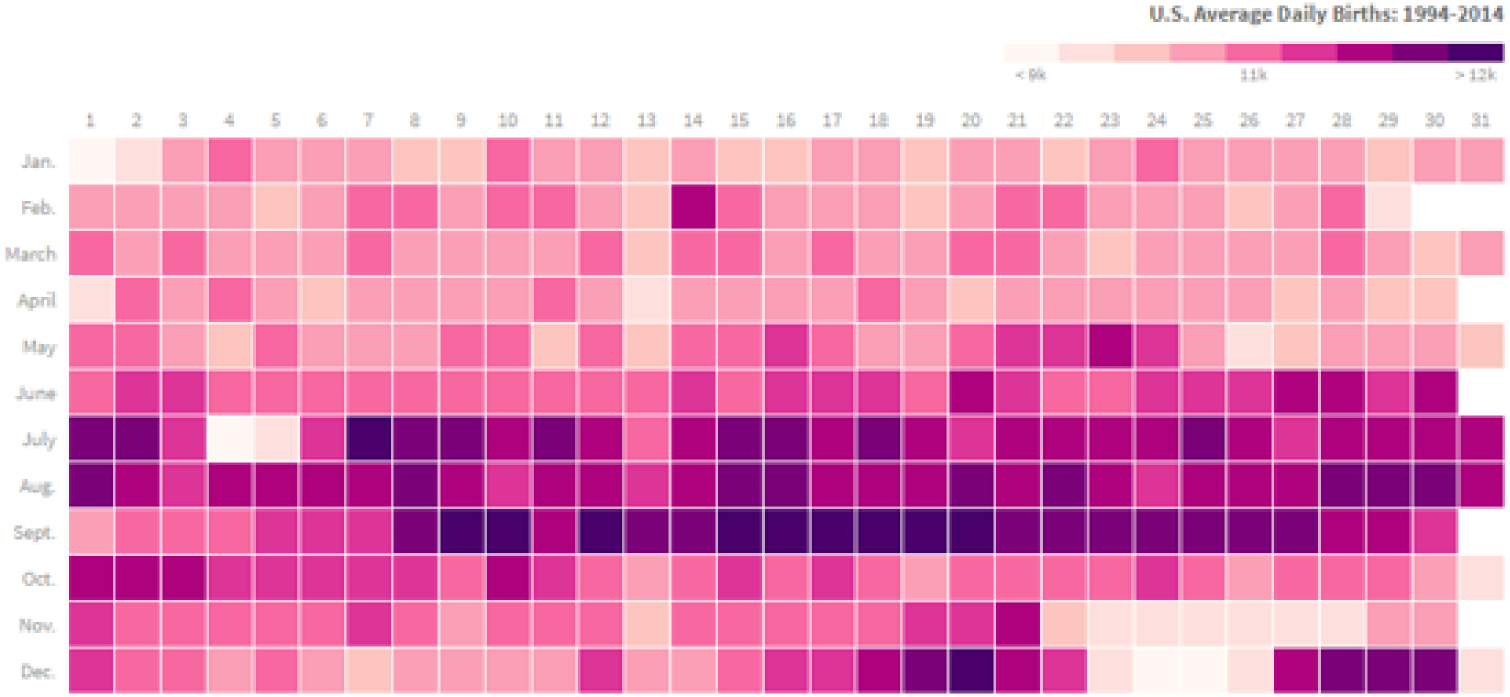
**Nacimientos registrados según mes de ocurrencia**

# Nacimientos registrados según mes de ocurrencia



# How Popular Is Your Birthday?

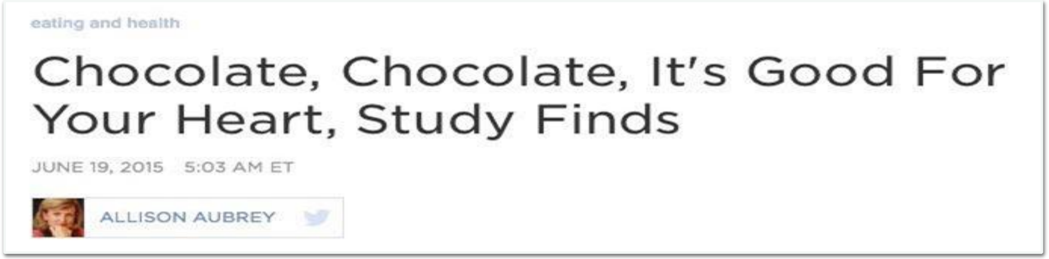
Two decades of American birthdays, averaged by month and day.



U.S. Average Daily Births: 1994-2014

https://visme.co/blog/most-common-birthday/

# Really?



eating and health

Chocolate, Chocolate, It's Good For Your Heart, Study Finds

JUNE 19, 2015 5:03 AM ET

ALLISON AUBREY

npr.org (report on a study in heart.bmj.com)

# Observation

- **individuals**, study subjects, participants, units
  - *European adults*

- **treatment**
  - *chocolate consumption*

- **outcome**
  - *heart disease*

# The first question

Is there any relation between chocolate consumption and heart disease?

- **association**

  “any relation”

# An answer

## Some data:

“Among those in the top tier of chocolate consumption, 12 percent developed or died of cardiovascular disease during the study, compared to 17.4 percent of those who didn’t eat chocolate.”

- *Howard LeWine of Harvard Health Blog, reported by [npr.org]*

- Yes, this points to an association

  (in my opinion)

# The next question

Does chocolate consumption lead to a reduction in heart disease?

- **causality**

This question is often harder to answer.

“[The study] doesn’t prove a cause-and-effect relationship between chocolate and reduced risk of heart disease and stroke.”

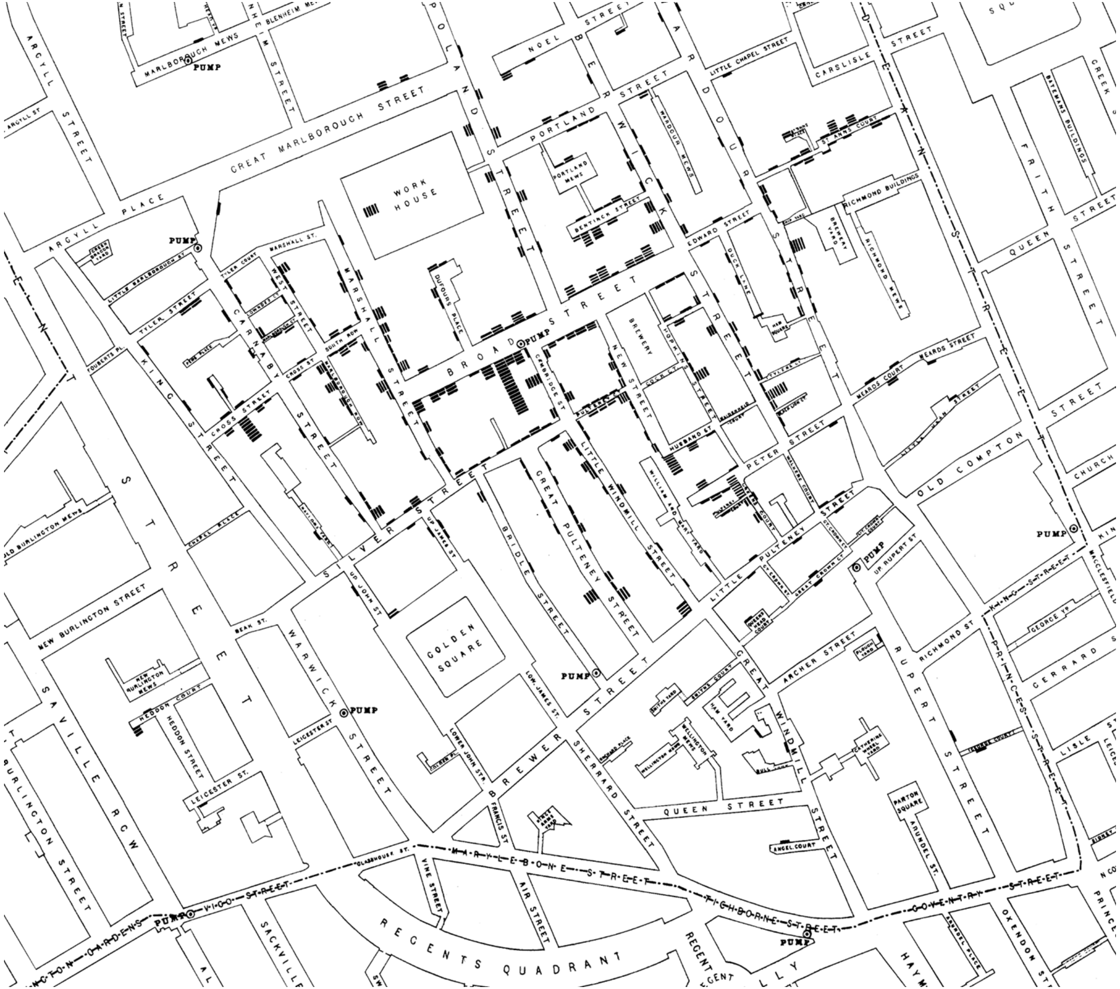
- *JoAnn Manson, chief of Preventive Medicine at Brigham and Women’s Hospital, Boston*

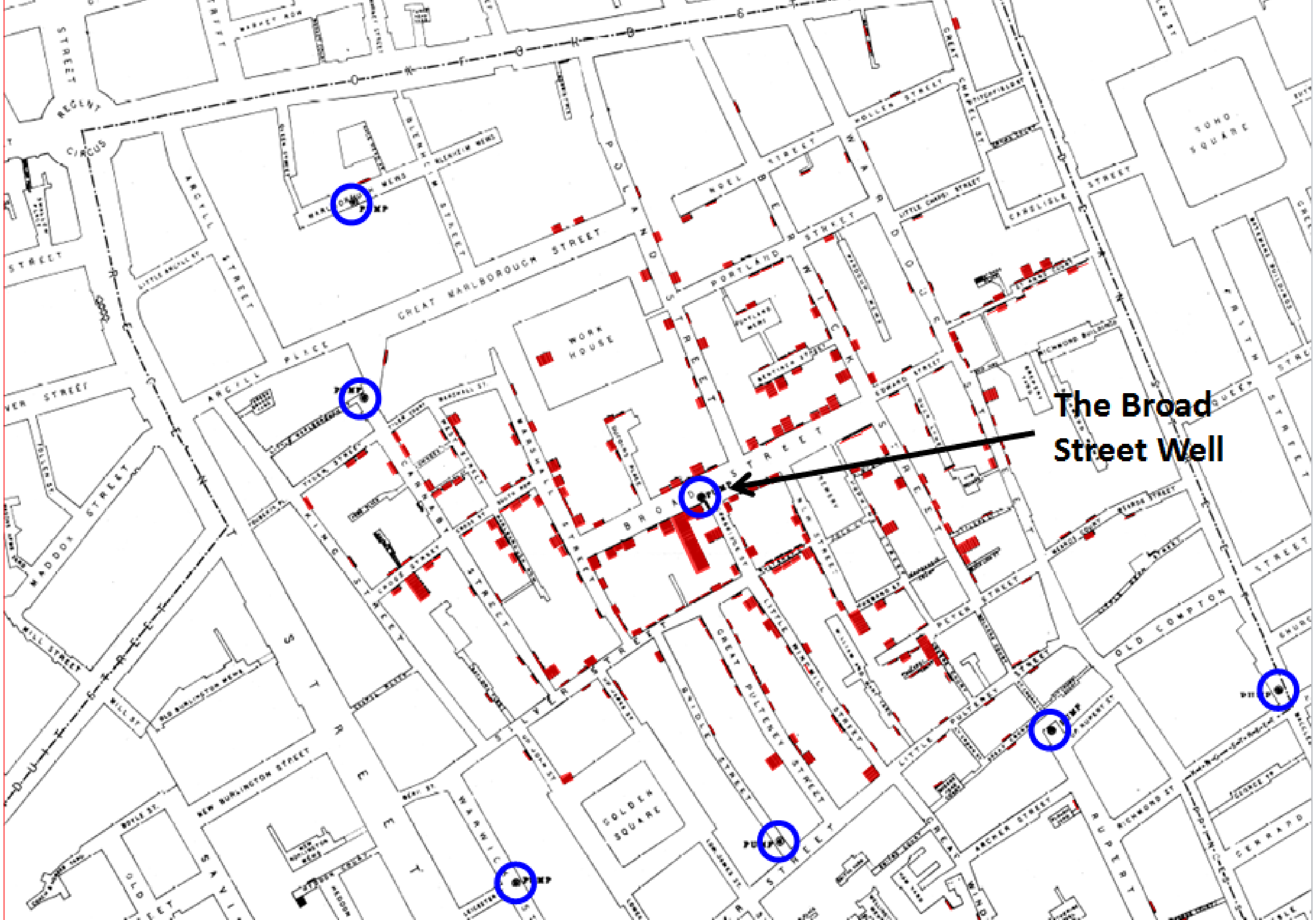
# Third cholera pandemic (1846–60)

# Miasmas, miasmatism, miasmatists

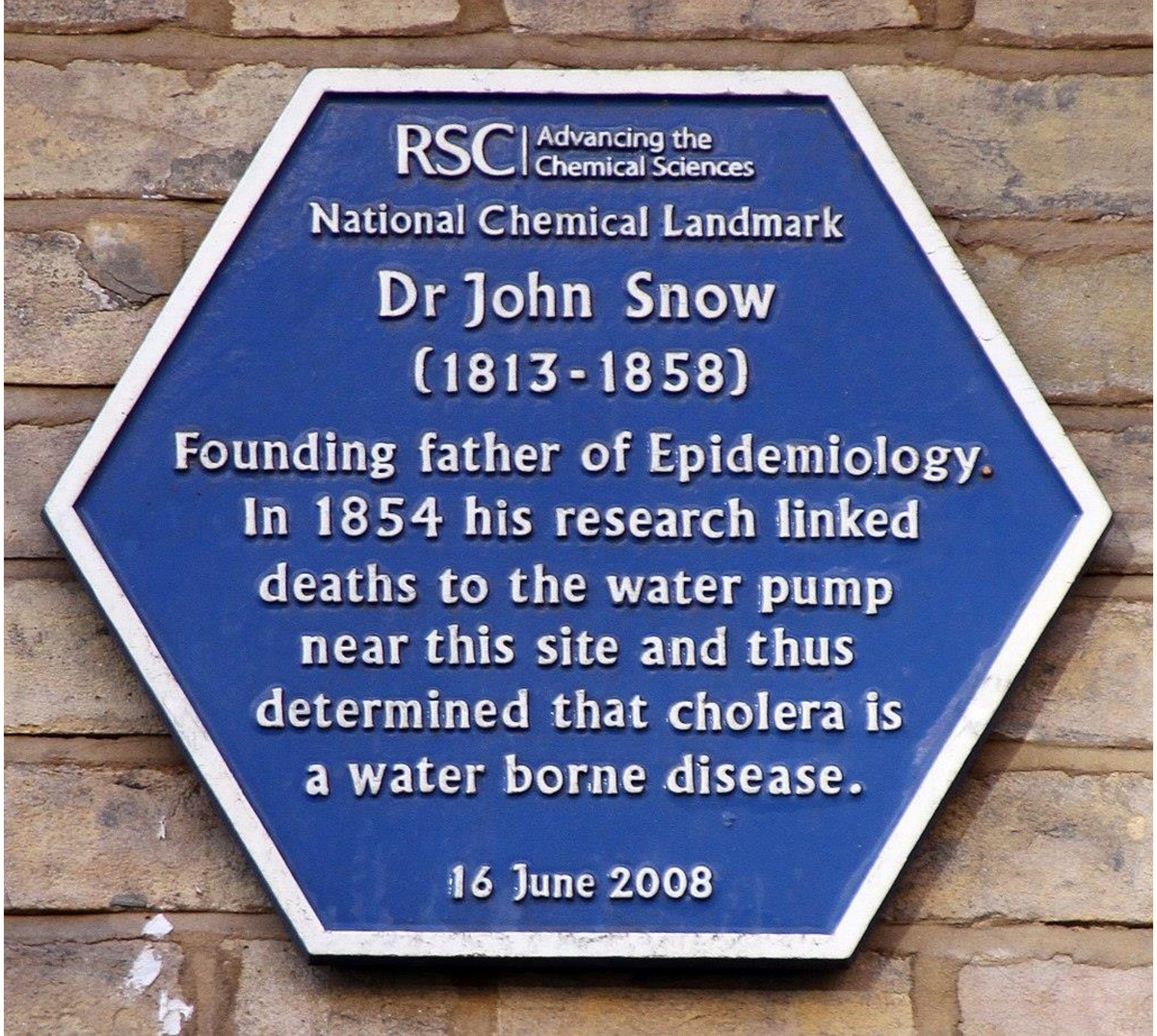
- **Bad smells** given off by waste and rotting matter
- **Believed to be the main source of disease**
- Suggested remedies:
  - “fly to clean air”
  - “fire off barrels of gunpowder”
- Staunch believers:
  - Florence Nightingale
  - Edwin Chadwick, Commissioner of the General Board of Health

# John Snow, 1813-1858


ARGYLL STREET
ARGYLL ST
MARLBOROUGH MEWS
PUMP
BLENHEIM MEWS
BLENHEIM STREET
GREAT MARLBOROUGH STREET
POLAND STREET
NOEL STREET
BERWICK STREET
WARDOUR STREET
WARDOUR MEWS
PORTLAND STREET
PORTLAND MEWS
BENTINCK STREET
LITTLE CHAPEL STREET
ST ANNS COURT
CARLISLE STREET
DEAN STREET
FRITH STREET
BATEMANS BUILDINGS
GREEK STREET
RICHMOND BUILDINGS
RICHMOND MEWS
BREWERY YARD
EDWARD STREET
DUCK LANE
HAM YARD
WORK HOUSE
ARGYLL PLACE
PUMP
GREEN DRAGON YARD
LITTLE MARLBOROUGH ST
TYLER COURT
MARSHALL ST.
MARSHALL STREET
WEST STREET
DUFOURS PLACE
TYLER STREET
CARNABY STREET
SOUTH ROW
BROAD STREET
PUMP
CAMBRIDGE ST
BREWERY
NEW STREET
HOPKINS STREET
QUEEN STREET
MEARDS COURT
MEARDS STREET
FOUBERTS PL.
KING STREET
CROSS STREET
CROSS ST.
PULTENEY COURT
HUSBAND ST
PETER STREET
NEW STREET
OLD COMPTON STREET
CHURCH STREET
PUMP
KING STREET
LITTLE WINDMILL STREET
GREAT PULTENEY STREET
WILLIAM AND MARY YARD
NEW STREET
SILVER STREET
OLD BURLINGTON MEWS
CHAPEL PLACE
UP. JAMES ST
UP. JOHN ST
BRIDLE STREET
LITTLE PULTENEY STREET
GREAT CROWN CT.
PUMP
UP. RUPERT ST
MACCLESFIELD STREET
REGENT STREET
NEW BURLINGTON STREET
BEAK ST.
WARWICK STREET
GOLDEN SQUARE
PUMP
BREWER STREET
GREAT WINDMILL STREET
ARCHER STREET
PLOUGH YARD
RUPERT STREET
RICHMOND ST
PRINCES STREET
GEORGE YD
GERRARD STREET
NEW BURLINGTON MEWS
HEDDON COURT
HEDDON STREET
SAVILLE ROW
PUMP
LEICESTER ST
LEICESTER ST.
LOW. JAMES ST
LOWER JOHN STR.
SMITHS COURT
HAM YARD
SHERRARD STREET
WELLINGTON MEWS
WELLINGTON PLACE
CATHERINE WHEEL YARD
LISLE STREET
BURLINGTON STREET
FRANCIS ST
QUEEN STREET
RUPERT STREET
ANGEL COURT
PANTON SQUARE
ARUNDEL ST.
GLASSHOUSE ST.
MARYLEBONE STREET
VINE STREET
AIR STREET
TICHBORNE STREET
COVENTRY STREET
PUMP
VIGO STREET
BURLINGTON GARDENS
SACKVILLE
RECENTS QUADRANT
REGENT
PICCADILLY
PUMP
HAYMARKET
OXENDON ST
PRINCES



The Broad Street Well



RSC | Advancing the Chemical Sciences

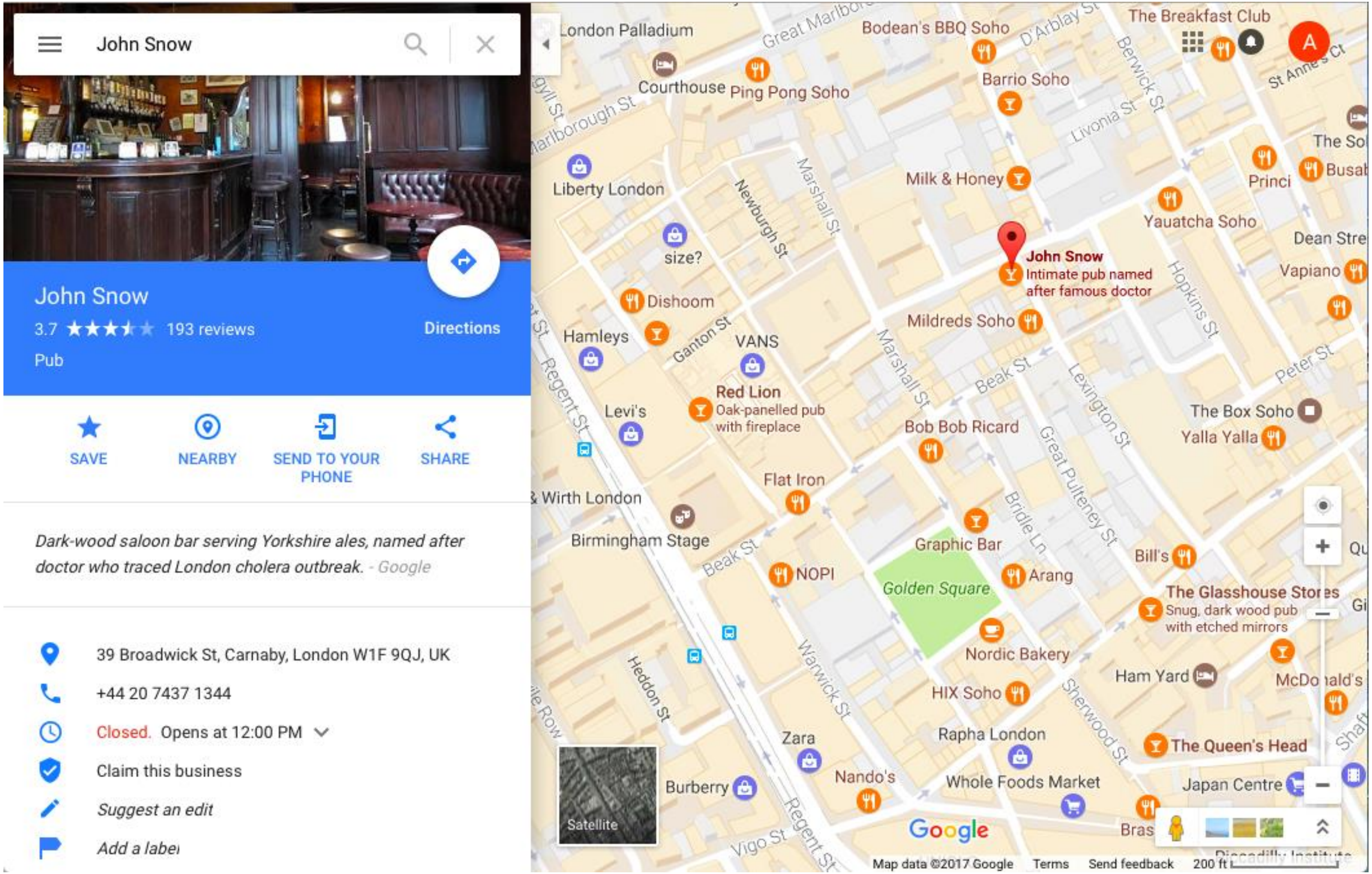
National Chemical Landmark

Dr John Snow

(1813-1858)

Founding father of Epidemiology.
In 1854 his research linked
deaths to the water pump
near this site and thus
determined that cholera is
a water borne disease.

16 June 2008



John Snow

John Snow

3.7 ★★★★☆ 193 reviews

Pub

*Dark-wood saloon bar serving Yorkshire ales, named after doctor who traced London cholera outbreak.* - Google

39 Broadwick St, Carnaby, London W1F 9QJ, UK

+44 20 7437 1344

Closed. Opens at 12:00 PM

Claim this business

*Suggest an edit*

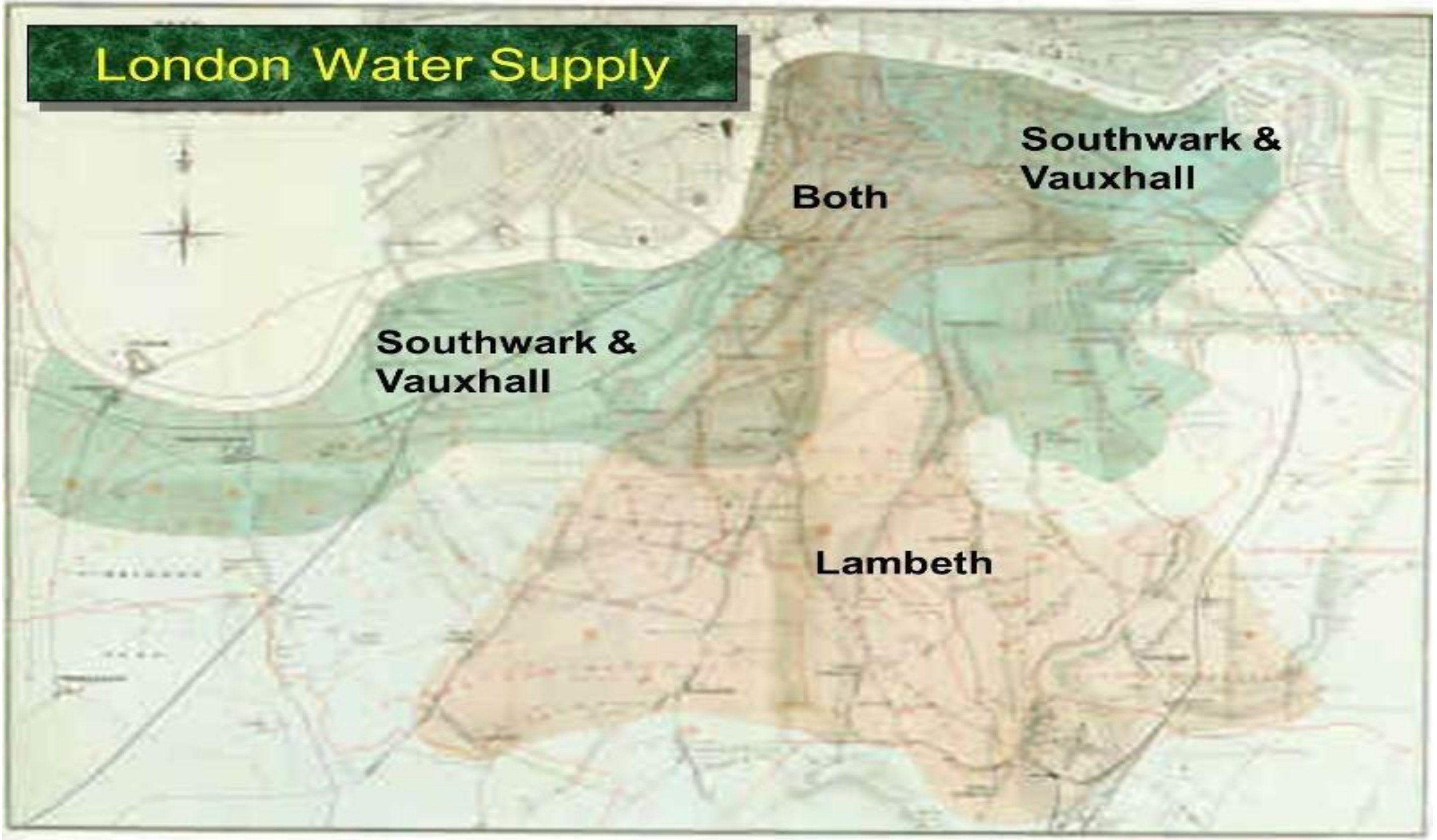
*Add a label*

LEXINGTON STREET
JOHN SNOW
JOHN SNOW
SALOON
SALOON
HN SNOW

# Comparison

- **treatment group**
- **control group**
  - does not receive the treatment

# London Water Supply



# Snow’s “Grand Experiment”

“… there is no difference whatever in the houses or the people receiving the supply of the two Water Companies, or in any of the physical conditions with which they are surrounded …”

- The two groups were *similar except for the treatment*.

# Snow’s table

| Supply Area | Number of houses | Cholera deaths | Deaths per 10,000 houses |
|---|---|---|---|
| S&V | 40,046 | 1,263 | 315 |
| Lambeth | 26,107 | 98 | 37 |
| Rest of London | 256,423 | 1,422 | 59 |

# Key to establishing causality

If the treatment and control groups are *similar apart from the treatment*, then differences between the outcomes in the two groups can be ascribed to the treatment.

# Trouble

If the treatment and control groups have systematic differences other than the treatment, then it might be difficult to identify causality.

Such differences are often present in **observational studies.**

When they lead researchers astray, they are called confounding factors.

# Different Type of Studies

- *Observational study*: the researcher does not choose which subjects receive the treatment

- *Controlled experiment*: the researcher designs a procedure for selecting the treatment and control groups

# Randomize!

- If you assign individuals to treatment and control **at random,** then the two groups are likely to be similar apart from the treatment.

- You can account – mathematically – for variability in the assignment.

- **Randomized Controlled Experiment**

# Careful ...

Regardless of what the dictionary says,
in probability theory

**Random ≠ Haphazard**

# Expressions

# Programming Languages

- Python is popular both for data science & general software development
- Mastering the language fundamentals is critical
- Learn through practice, not by reading or listening

(Demo)

# Arithmetic Operators

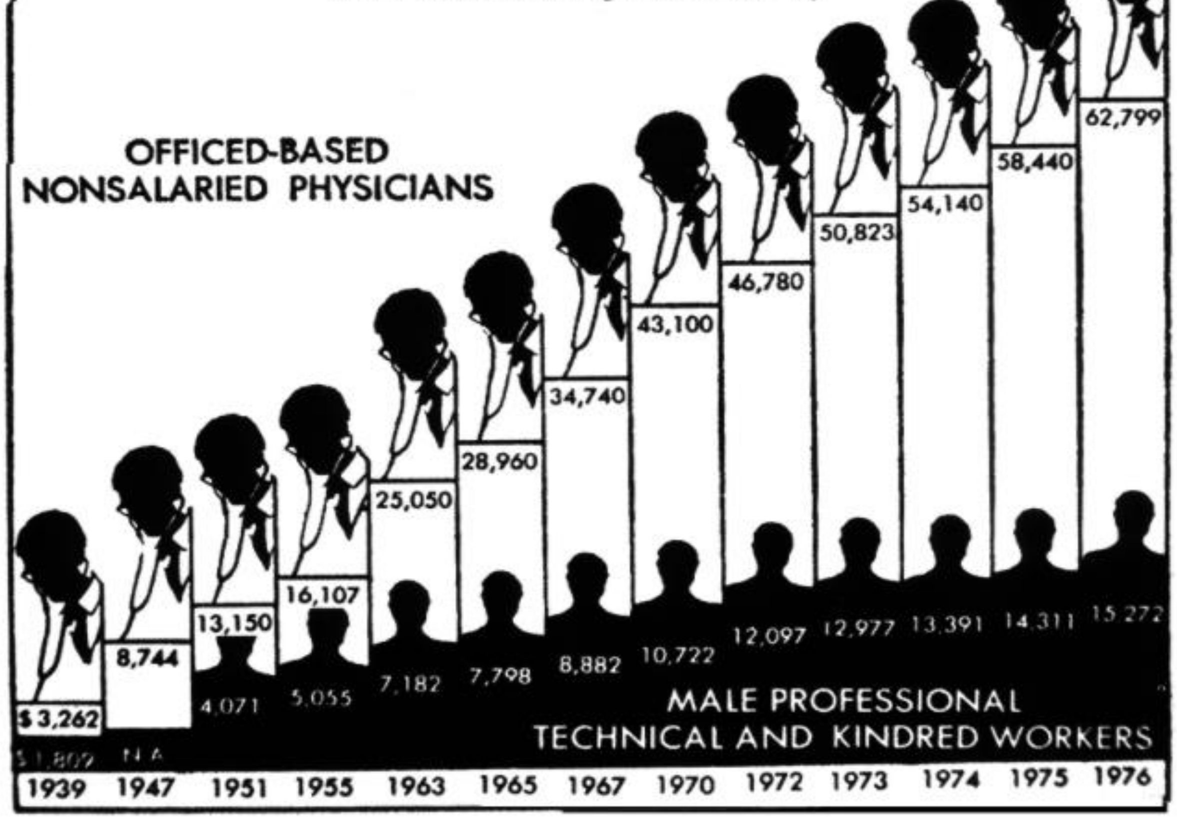
| Operation | Operator | Example | Value |
|---|---|---|---|
| Addition | + | 2 + 3 | 5 |
| Subtraction | - | 2 - 3 | -1 |
| Multiplication | * | 2 * 3 | 6 |
| Division | / | 7 / 3 | 2.66667 |
| Remainder | % | 7 % 3 | 1 |
| Exponentiation | ** | 2 ** 0.5 | 1.41421 |

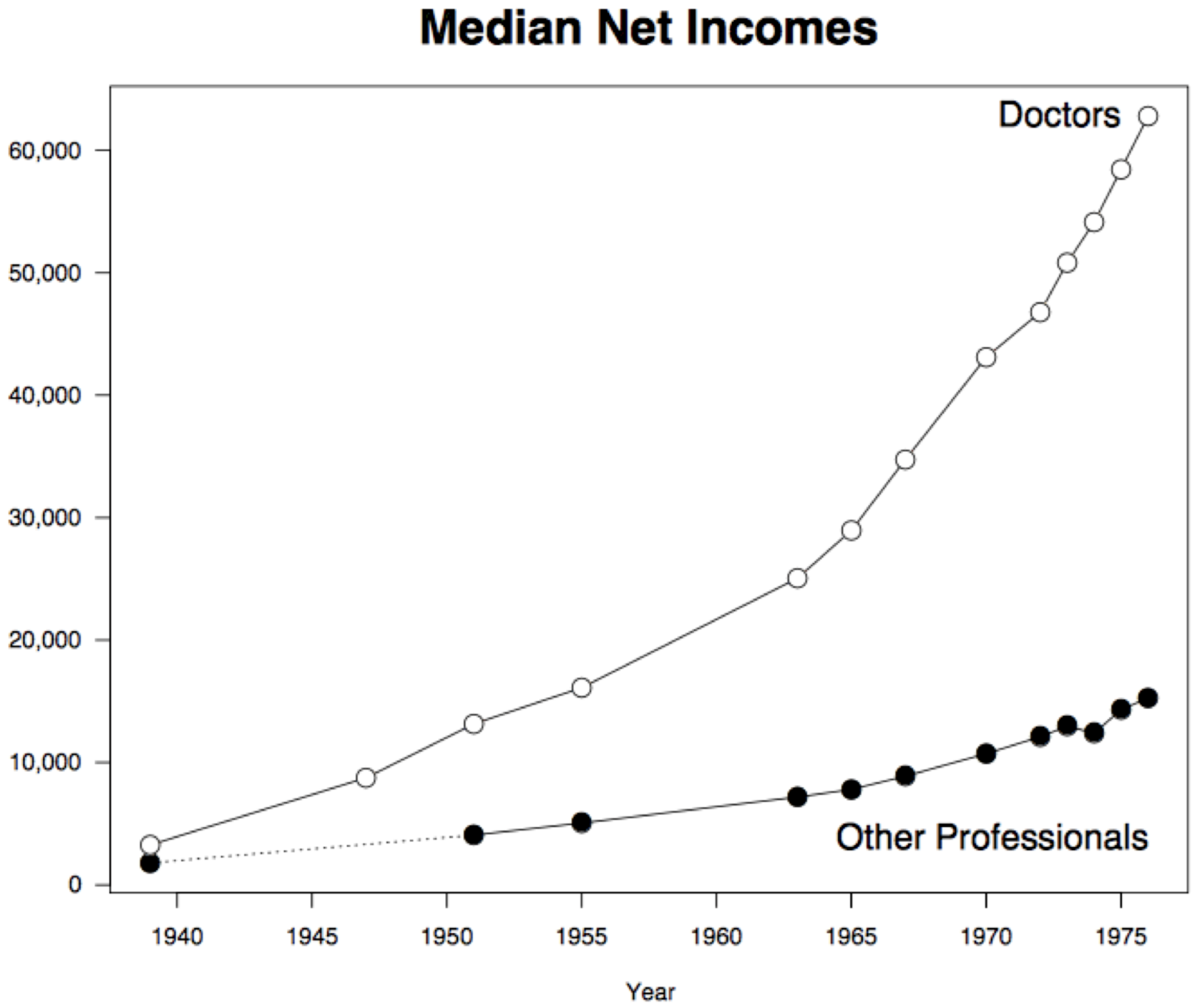
# Example: Slopes

# Incomes of Doctors Vs. Other Professionals

(MEDIAN NET INCOMES)

**SOURCE:** Council on Wage and Price Stability




Median Net Incomes
Doctors
Other Professionals
60,000
50,000
40,000
30,000
20,000
10,000
0
1940
1945
1950
1955
1960
1965
1970
1975
Year

(Demo)

# Numbers

(Demo)

# Ints and Floats

Python has two real number types

- **`int`**: an integer of any size
- **`float`:** a number with an optional fractional part

An **`int`** never has a decimal point; a **float** always does

A **`float`** might be printed using scientific notation

Three limitations of float values:

- They have limited size (but the limit is huge)
- They have limited precision of 15-16 decimal places
- After arithmetic, the final decimal few places can be wrong

# Discussion Question

Rank the results of the following expressions in order from least to greatest

A. `3 * 10 ** 10`

B. `10 * 3 ** 10`

C. `(10 * 3) ** 10`

D. `10 / 3 / 10`

E. `10 / (3 / 10)`

A. `30000000000`

B. `590490`

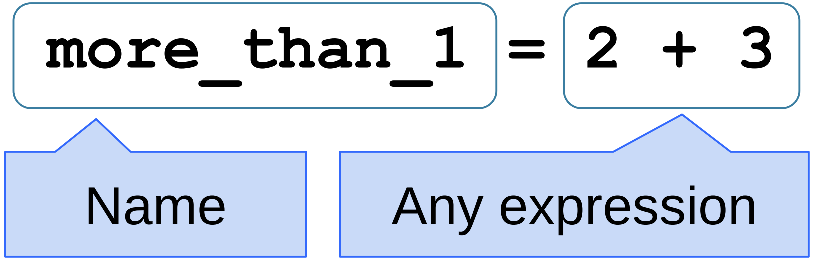
C. `590490000000000`

D. `0.33333333333333337`

E. `33.333333333333336`

# Names

# Assignment Statements



```
more_than_1 = 2 + 3
```

Name

Any expression

- Statements don't have a value; they perform an action
- An assignment statement changes the meaning of the name to the left of the = symbol
- The name is bound to a value (not an equation)

(Demo)

# Exponential Growth

# Growth Rate

- The rate of increase per unit time
- After one time unit, a quantity **`x`** growing at rate **`g`** will be

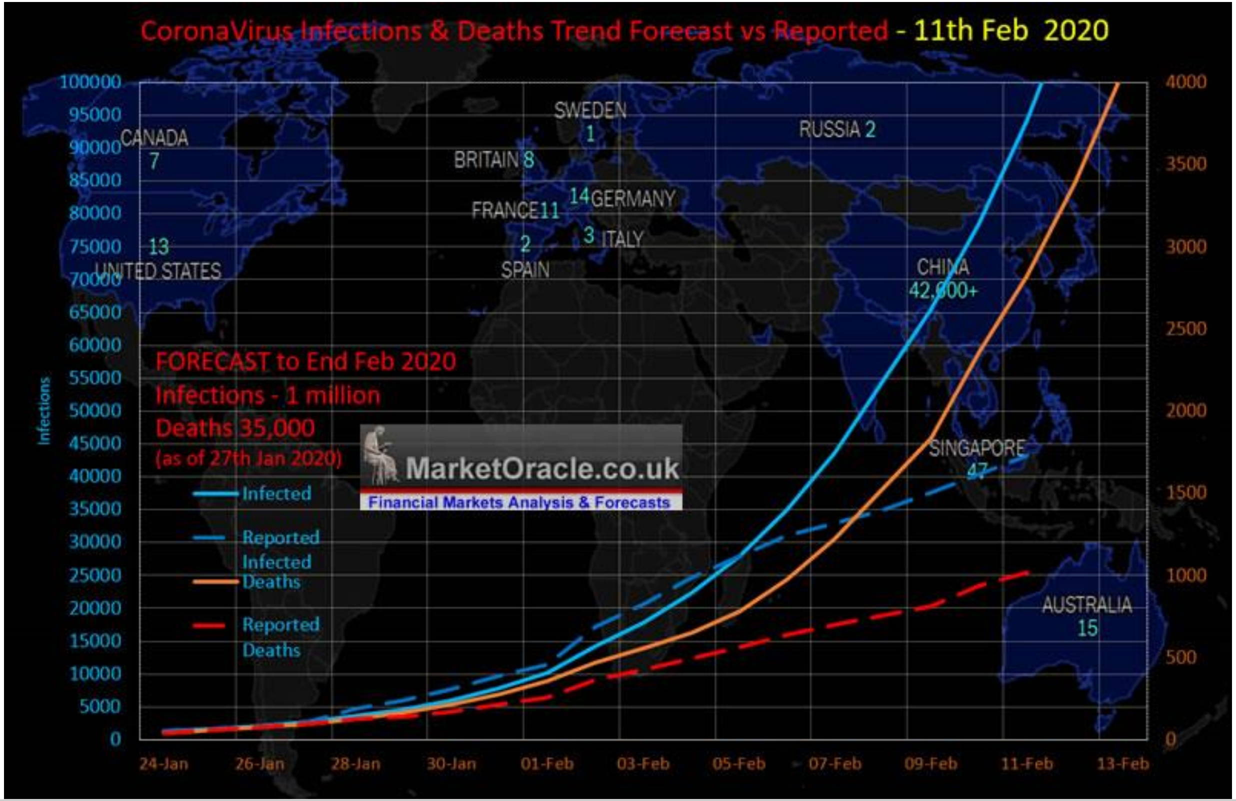
  **`x * (1 + g)`**
- After **`t`** time units, a quantity **`x`** growing at rate **`g`** will be

  **`x * (1 + g) ** t`**
- If **`after`** and **`before`** are measurements of the same quantity taken **`t`** time units apart, then the growth rate is

  **`(after/before) ** (1/t) - 1`**

(Demo)

# CoronaVirus-2020



CoronaVirus Infections & Deaths Trend Forecast vs Reported - 11th Feb 2020

FORECAST to End Feb 2020

Infections - 1 million

Deaths 35,000

(as of 27th Jan 2020)

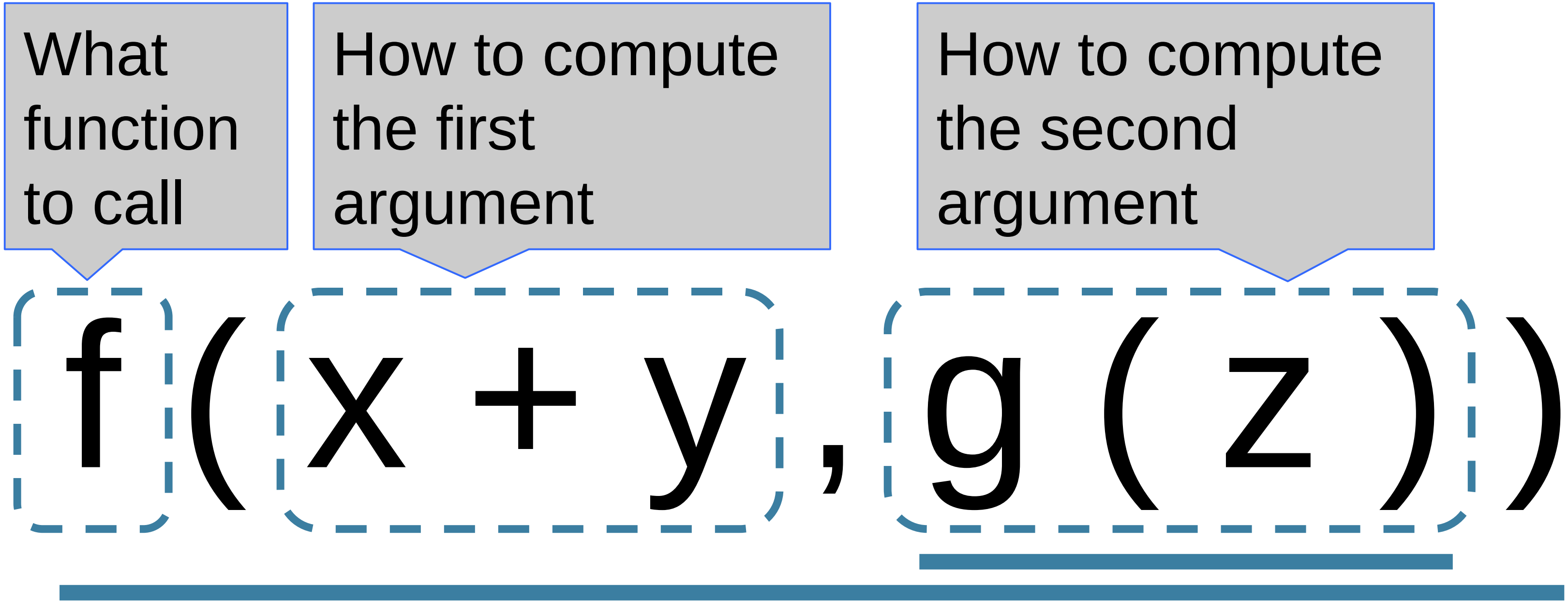
MarketOracle.co.uk

Financial Markets Analysis & Forecasts

# Call Expressions

(Demo)

# Anatomy of a Call Expression



What function to call

How to compute the first argument

How to compute the second argument

```
f(x + y, g(z))
```

"Call f on the result of adding x + y and the return value of calling g on z."

# Discussion Question

Assume you have run the following statements

```
x = 3
y = -2.0
```

Which of these examples results in an error?

A. `abs(x, y)`

B. `math.pow(x, abs(y))`

C. `round(x, max(abs(y ** 2))))`

D. `math.pow(x, math.pow(y, x))`

# Strings

# Text and Strings

A string value is a snippet of text of any length

- `'a'`
- `'word'`
- `"there can be 2 sentences. Here's the second!"`

Strings that contain numbers can be converted to numbers

- `int('12')`
- `float('1.2')`

Any value can be converted to a string

- `str(5)`

(Demo)

# Discussion Question

Assume you have run the following statements

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

What's the source of the error in each example?

A. `x + y`

B. `x + int(y + z)`

C. `str(x) + int(y)`

D. `str(x, y) + z`

# A Note on Functions/Methods

- Functions are called by themselves:

```
abs(-2)
int('42')
```

- Methods are tied to a particular type:

```
'hello'.count(2)
'Sam is kinda cool'.replace('kinda', 'very')
math.pow(2, 5)
```

# Arrays

# Arrays

An array contains a sequence of values

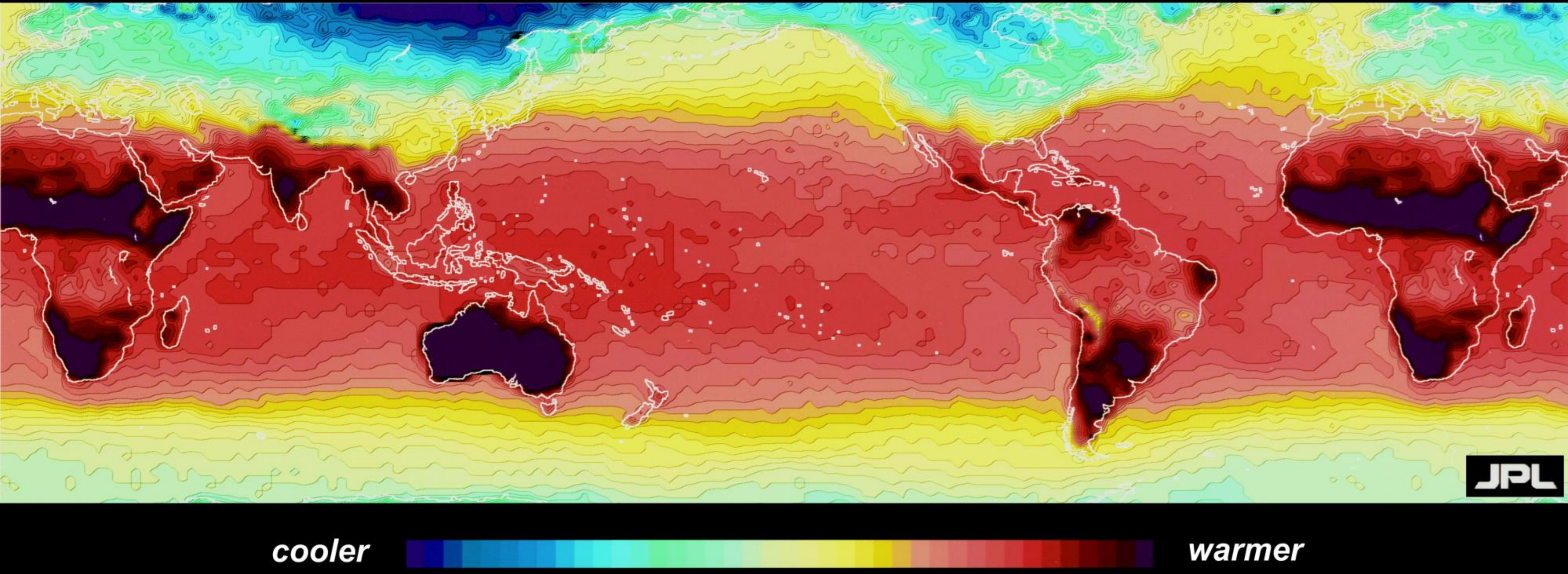
- All elements of an array should have the same type
- Arithmetic is applied to each element individually
- When two arrays are added, they must have the same size; corresponding elements are added in the result

http://berkeleyearth.lbl.gov/regions/global-land

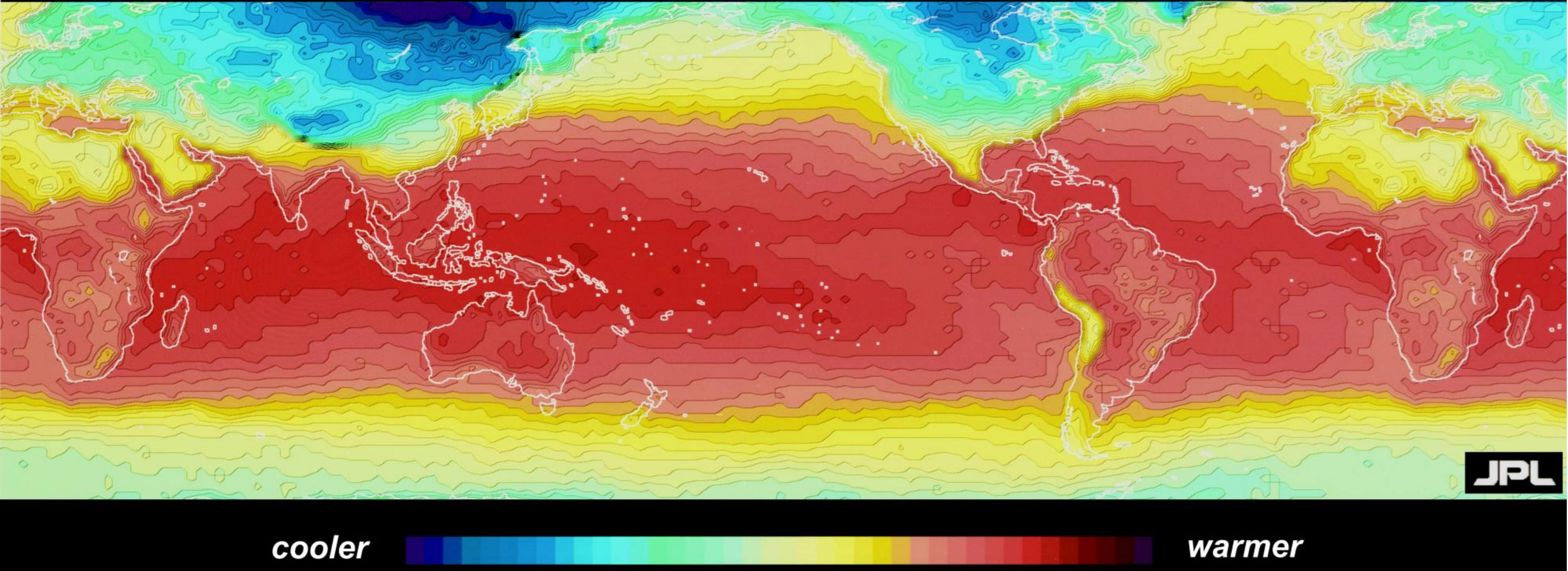
(Demo)

http://www.meteo.psu.edu/~j2n/Ed5_Fig3_6a.JPG vs http://www.meteo.psu.edu/~j2n/Ed5_Fig3_6b.JPG a)

## Mean Daytime Surface Temperature (January 1979)



b) **Mean Nighttime Surface Temperature (January 1979)**



# Ranges

# 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.arange(start, end, step)`**:
  A range with **`step`** between consecutive values

The range always includes **`start`** but excludes **`end`**

(Demo)

Leibniz formula for π

# Discussion Question

Assume you have run the following statements

```
x = make_array(2, 3, 4)
y = np.arange(2, 3, 4)
z = np.arange(3)
```

Which lines error?

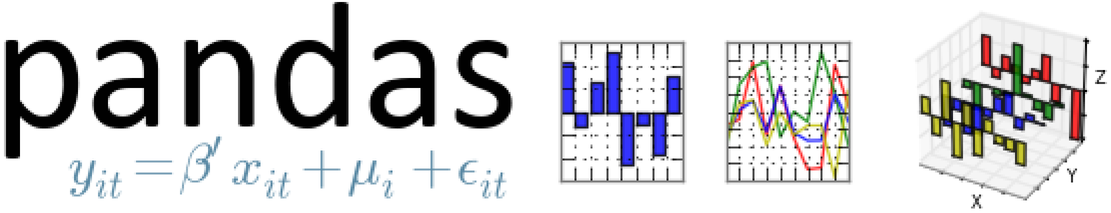
A. `x + y`

B. `x + z`

C. `x.item(0) + y.item(0)`

D. `x.item(1) + y.item(1)`

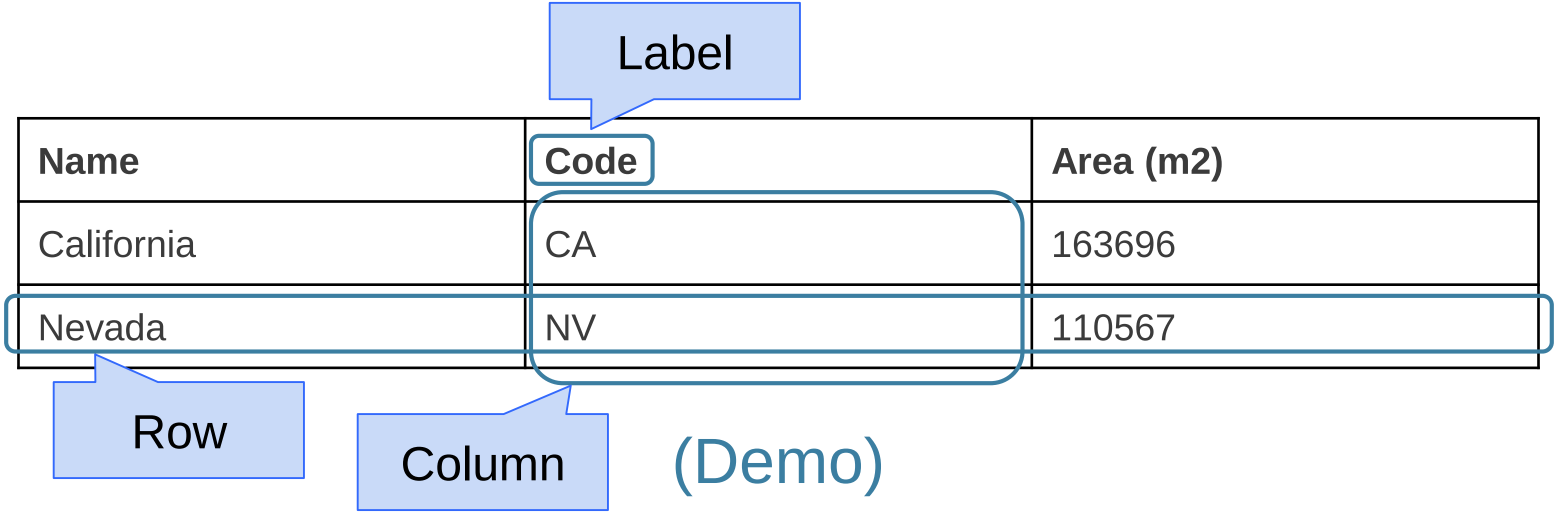
# Tables



Python Data Analysis Library

# Table Structure

- A Table is a sequence of labeled columns
- Labels are strings
- Columns are arrays, all with the same length



Label

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

Row

Column

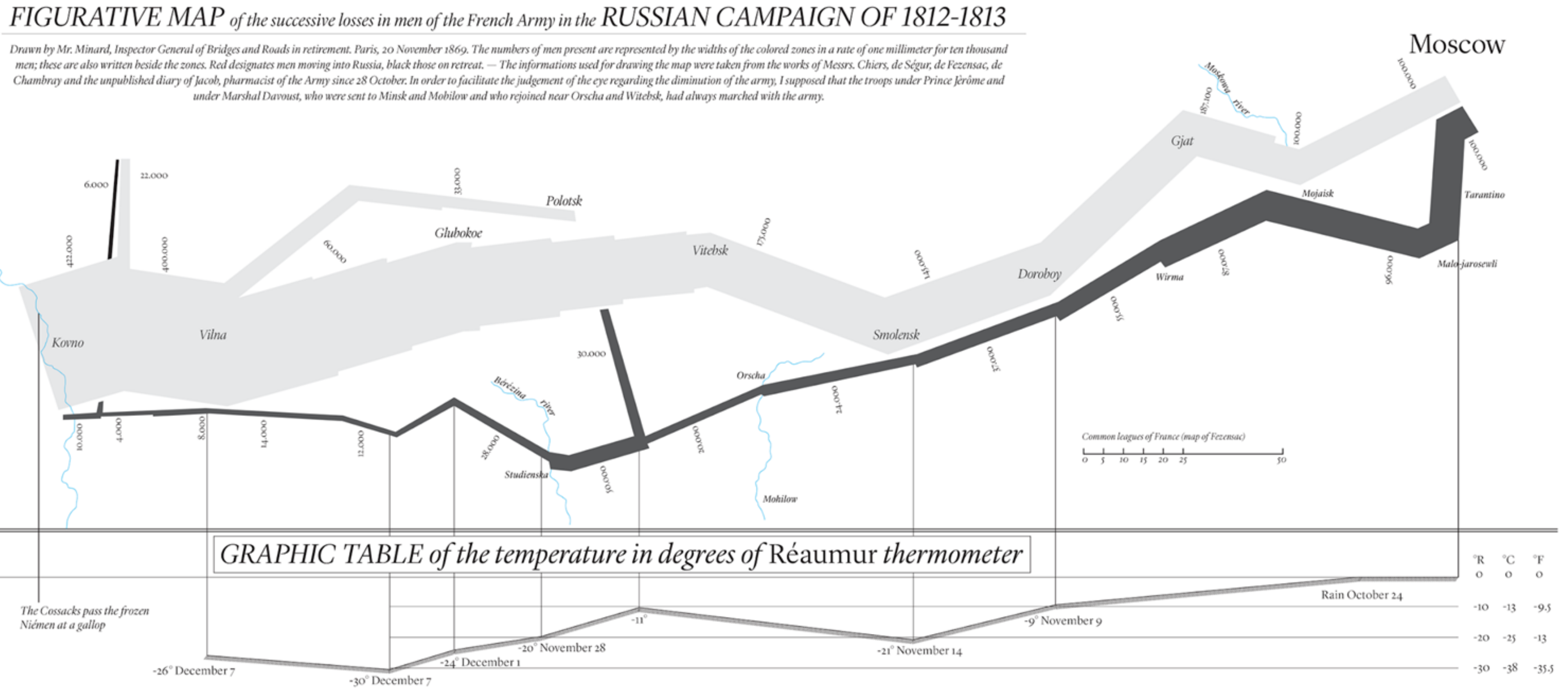
(Demo)

# Minard's Map

# Charles Joseph Minard, 1781-1870

- French civil engineer who created one of the greatest graphs of all time
- Visualized Napoleon's 1812 invasion of Russia, including
  - the number of soldiers
  - the direction of the march
  - the latitude and longitude of each city
  - the temperature on the return journey
  - Dates in November and December

# Visualization of 1812 March



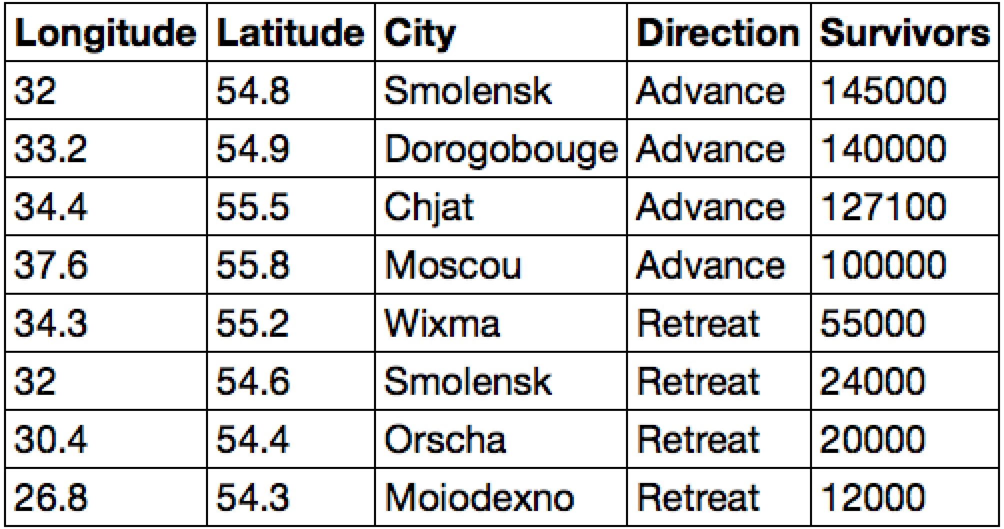
FIGURATIVE MAP *of the successive losses in men of the French Army in the* RUSSIAN CAMPAIGN OF 1812-1813

*Drawn by Mr. Minard, Inspector General of Bridges and Roads in retirement. Paris, 20 November 1869. The numbers of men present are represented by the widths of the colored zones in a rate of one millimeter for ten thousand men; these are also written beside the zones. Red designates men moving into Russia, black those on retreat. — The informations used for drawing the map were taken from the works of Messrs. Chiers, de Ségur, de Fezensac, de Chambray and the unpublished diary of Jacob, pharmacist of the Army since 28 October. In order to facilitate the judgement of the eye regarding the diminution of the army, I supposed that the troops under Prince Jérôme and under Marshal Davoust, who were sent to Minsk and Mobilow and who rejoined near Orscha and Witebsk, had always marched with the army.*

GRAPHIC TABLE *of the temperature in degrees of* Réaumur *thermometer*

*The Cossacks pass the frozen Niémen at a gallop*

# Different types of data



| Longitude | Latitude | City | Direction | Survivors |
|---|---|---|---|---|
| 32 | 54.8 | Smolensk | Advance | 145000 |
| 33.2 | 54.9 | Dorogobouge | Advance | 140000 |
| 34.4 | 55.5 | Chjat | Advance | 127100 |
| 37.6 | 55.8 | Moscou | Advance | 100000 |
| 34.3 | 55.2 | Wixma | Retreat | 55000 |
| 32 | 54.6 | Smolensk | Retreat | 24000 |
| 30.4 | 54.4 | Orscha | Retreat | 20000 |
| 26.8 | 54.3 | Moiodexno | Retreat | 12000 |

**float**:
decimal number

**string**:
text

**int**:
integer

# Sort

# Sorting Tables

Tables are ordered collections of rows

The `sort` method creates a new table with the same rows in a different order (the original table is unaffected)

The `show` method displays the first rows of a table

(Demo)

# Lists

# Lists are Generic Sequences

A list is a sequence of values (just like an array), but the values can all have different types

```
[2+3, 'four', Table().with_column('K', [3, 4])]
```

If you create a table column from a list, it will be converted to an array automatically

(Demo)

# Take

# Take Rows, Select Columns

The `select` method returns a table with only some columns

The `take` method returns a table with only some rows

- Rows are numbered, starting at 0
- Taking a single number returns a one-row table
- Taking a list of numbers returns a table as well

(Demo)

# Where

# The Where Method

The where method specifies a column and a condition

It returns a new table with all rows satisfying the condition

(Demo)

https://www.inferentialthinking.com/chapters/05/2/selecting-rows.html#Some-More-Conditions

# Manipulating Rows

- `t.sort(column)` sorts the rows in increasing order
- `t.take(row_numbers)` keeps the numbered rows
    - Each `row` has an index, starting at 0
- `t.where(`*`column`*`, are.`*`condition`*`)` keeps all rows for which a column's value satisfies a condition
- `t.where(`*`column`*`, value)` keeps all rows containing a certain value in a column