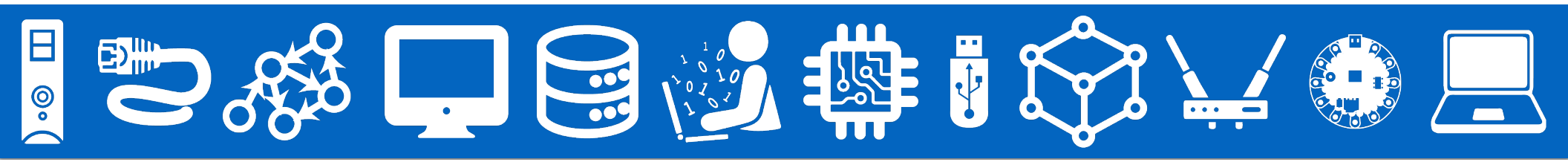
CS134:
Plotting with matplotlib



Announcements & Logistics

- **Lab 6** is today/tomorrow, due Wed/Thur
- **HW 6** will be posted on Wed (back to our "normal" schedule)
- We're working on the midterms and will return them ASAP
- Please fill out the **CS134 TA feedback form** by Friday

Do You Have Any Questions?

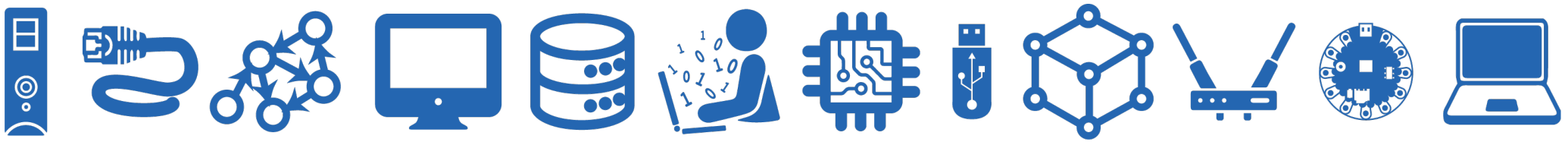
Last Time

- Wrapped up dictionaries
- Investigated **sorting** with dictionaries
- Discussed a new unordered data structure: **sets**
- Reviewed all data structures so far and when to use each

Today's Plan

- Learn about **plotting** with matplotlib
- Gain more practice using dictionaries, sets, tuples, and file reading
    - You'll gain more practice in lab this week

Plotting



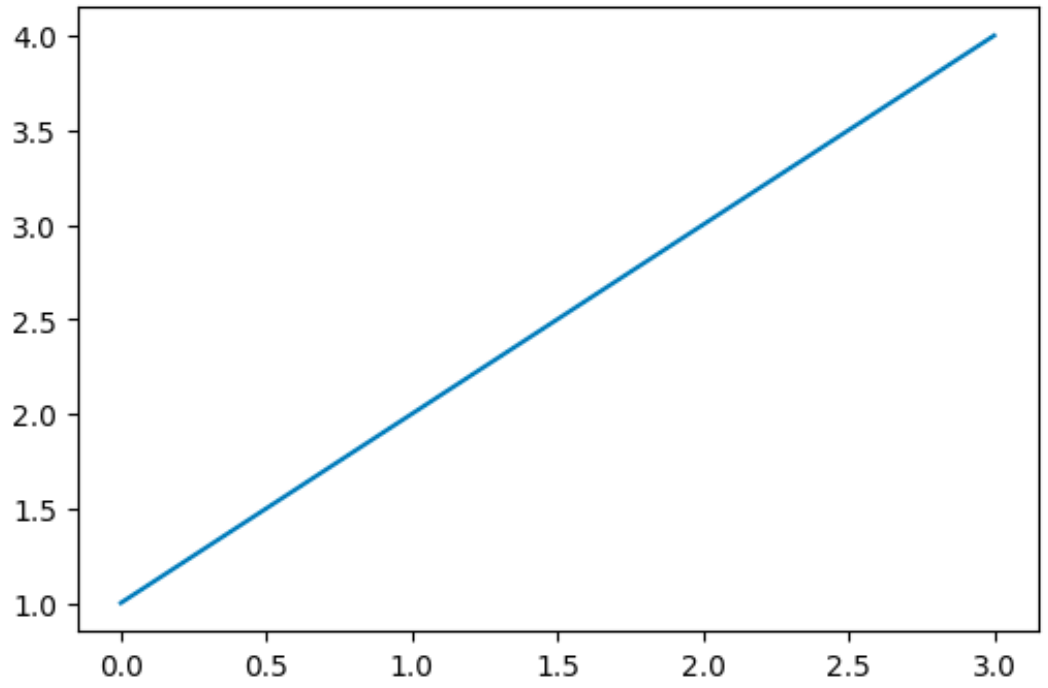
Plotting with `matplotlib`

- Suppose we want to a way to visualize our data (not just print it to the terminal)
- A plot is a graphical technique for representing a data set, usually as a graph showing the relationship between two or more variables
- We'll be using Python's `matplotlib` library to make plots/graphs
- The best way to learn how to plot different types of graphs is to read the documentation and see examples
- **Resources**
  - **matplotlib examples**: http://matplotlib.org/examples
  - **pyplot documentation**: http://matplotlib.org/api/pyplot_summary.html
  - **cool plots**: https://matplotlib.org/gallery.html

Plotting Basics: Plot function

```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.show()
```

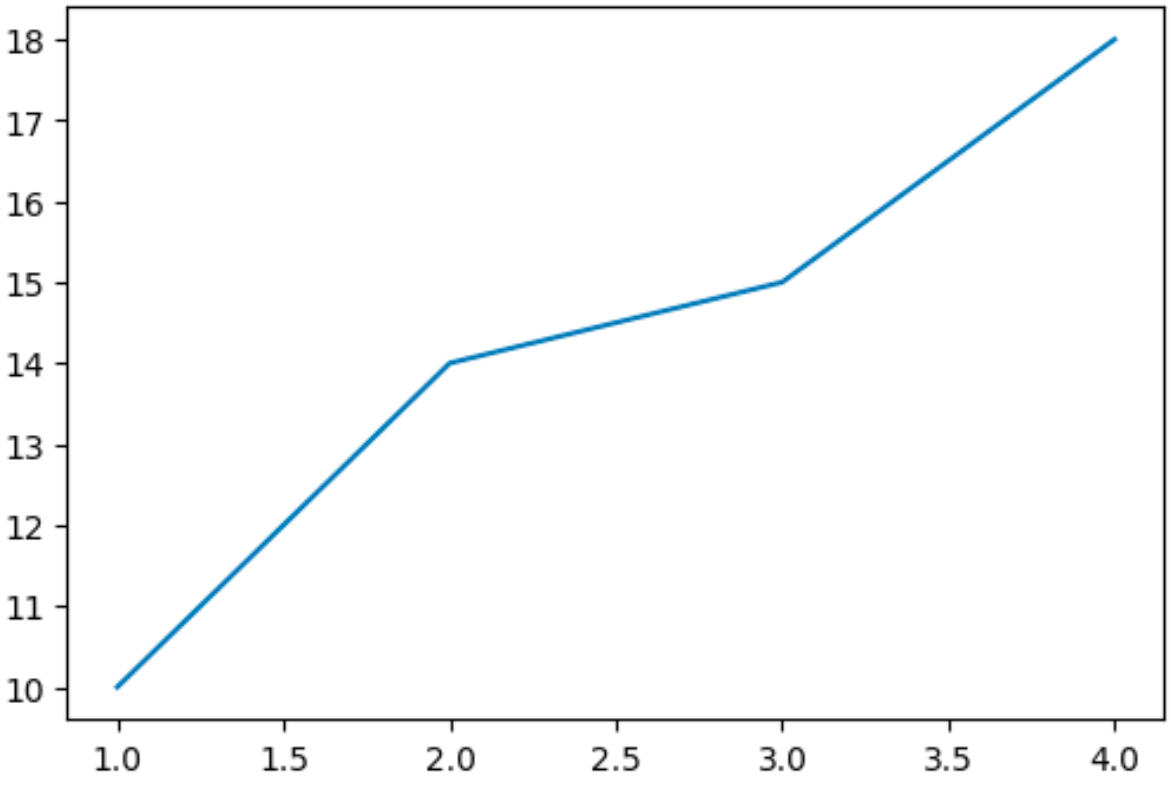
If only one list is provided, Python assumes it is as the points on the **y axis** (x values start at 0)



Plotting Basics: Plot function

```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4], [10, 14, 15, 18])
plt.show()
```

Equivalent to saying plot the points
`(1, 10), (2, 14), (3, 15), (4, 18)`



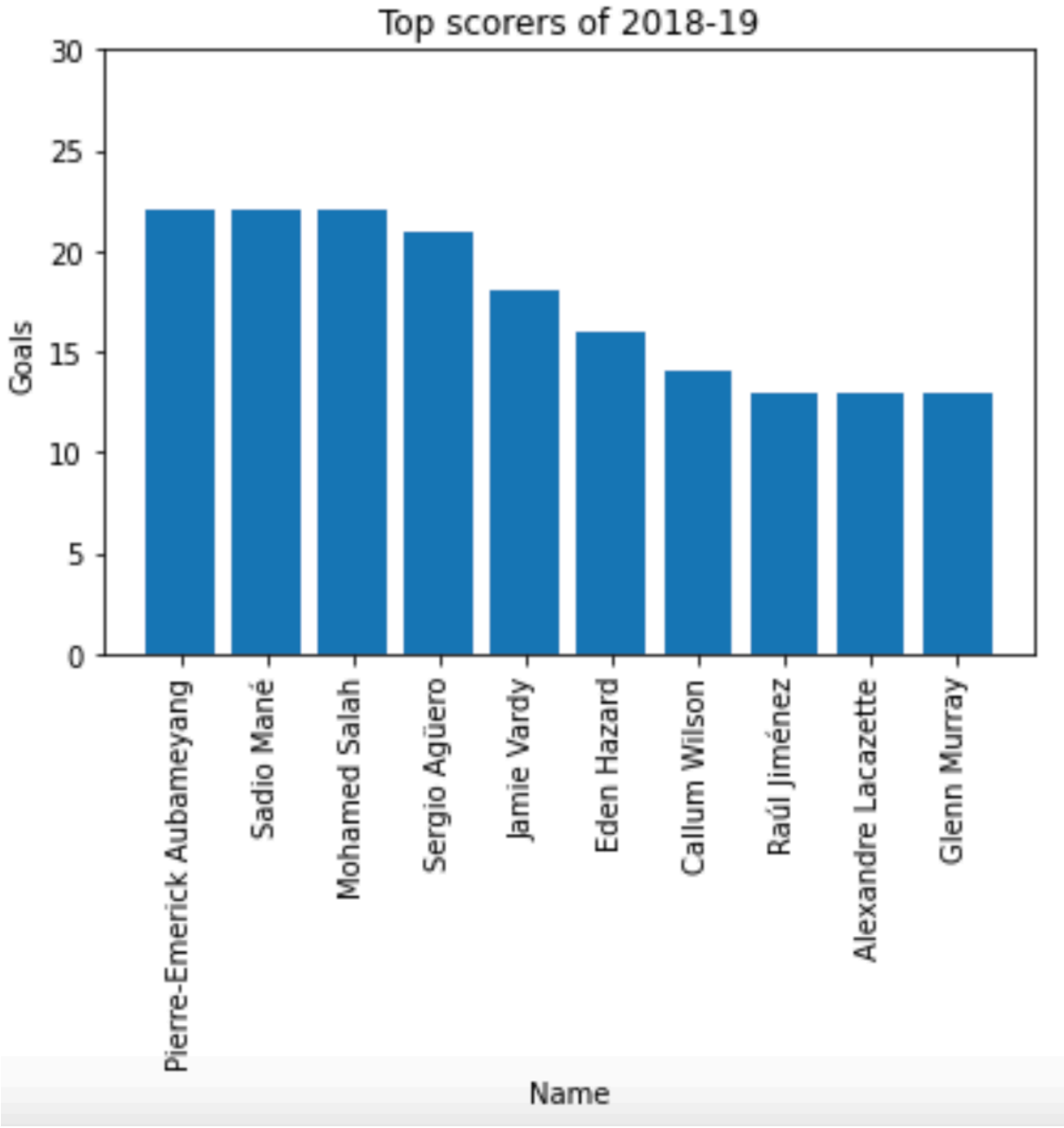
Exercise: Jupyter notebook

You are a talent scout for an English football (soccer) club. The club you work for has a good defense, but a weak offense. So, you've been tasked with identifying a star striker to help score more goals!

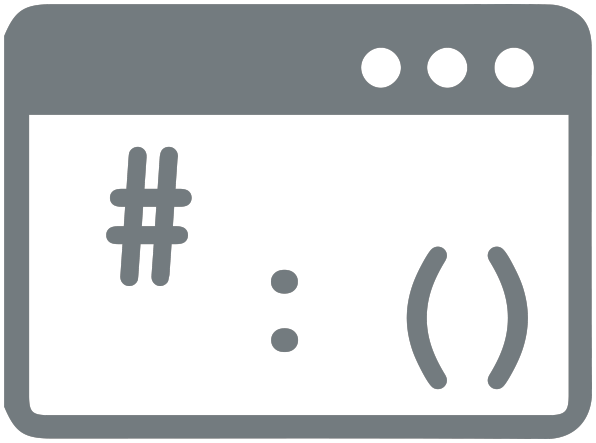
So you decide to identify candidates in a data-driven manner.

What we're aiming to produce

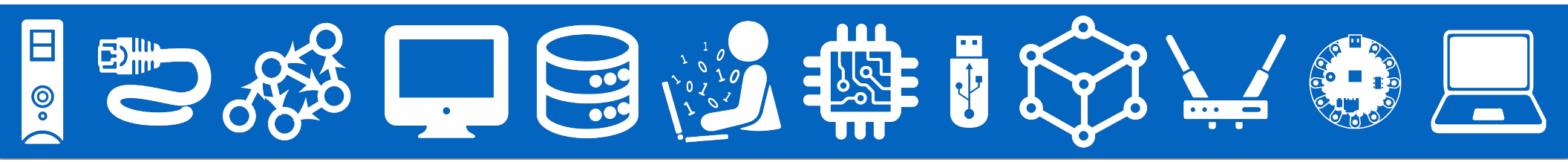
- We will plot bar charts showing the most frequent goal scorers in various years, and use them to determine who to try and recruit to our team



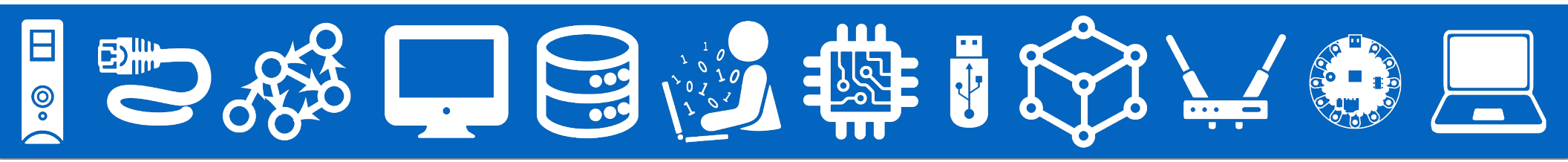
See:
Jupyter Notebook



The end!



CS134:
Lab 6

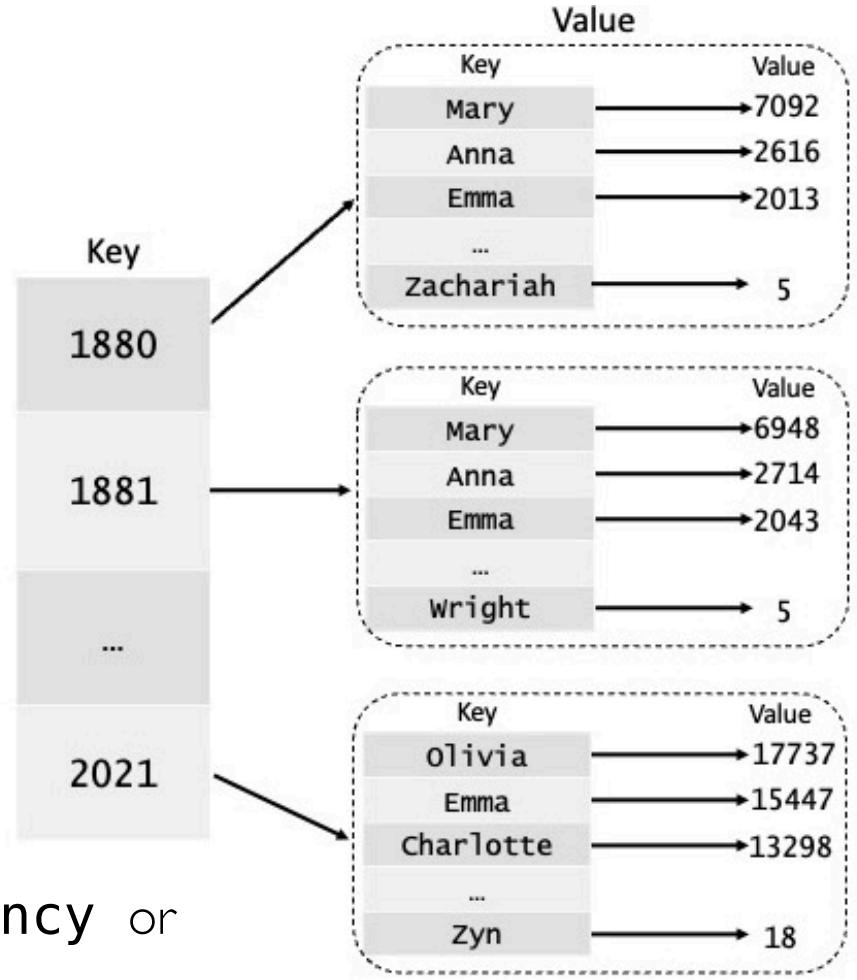


Lab 6 Overview

- Using data obtained from the US Social Security Administration about the popularity of names assigned to babies at birth, do some simple data analysis to determine:
    - The changing popularity of names over time
    - The changing popularity of the first initial of names over time
- In doing so, you will gain experience with the following:
    - Reading data from CSVs
    - Using dictionaries (and dictionaries of dictionaries)
    - Plotting different kinds of graphs with matplotlib

Dictionaries of Dictionaries

- Outer year dictionary maps integer years to "inner" name dictionaries
  - `nameDB = yearDB[1880]`
  - How to use `.get()` with `defaultVal` of an empty dictionary?
  - `nameDB = yearDB.get(1880, dict())`
- Inner dictionaries map string names to integer frequencies
  - `newFreq = nameDB["Mary"] + frequency` or
  - `newFreq = nameDB.get("Mary", 0) + frequency`
  - `nameDB["Mary"] = val`



readNames

1. Read file (line by line): name, year, sex, frequency
2. Get "inner" dictionary out of outer dictionary for year

```
nameDB = yearDB.get(year, dict())
```

3. Update inner dictionary for name (increment frequency)

```
newFreq = nameDB.get(name, 0) + frequency
nameDB[name] = newFreq
```

4. Update outer dictionary with updated inner dictionary

```
yearDB[year] = nameDB
```

Hints

- Pay close attention to data types required for keys and values in dictionaries
- Test your code often!
- Use print to investigate data structures as needed
- Be creative and have fun!