CS 134:
Plotting with matplotlib
Announcements & Logistics

• **Lab 5** is due Friday at noon
• No TA hours on Thursday evening
• Come see/email an instructor if necessary
• See calendar for office hours this week

• **Midterm**: Thu Mar 17th
  • Attend one slot: 6 - 7:30pm or 8 - 9:30pm in **Wachenheim B11**
  • **Wachenheim 002** at 6pm for reduced distractions/extra time

• **Practice midterm** on Glow
• 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
An Overview of Python Data Structures (so far!)
## Python Data Structures at a Glance

<table>
<thead>
<tr>
<th></th>
<th>Lists</th>
<th>Tuples</th>
<th>Dictionaries</th>
<th>Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Order</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Mutability</strong></td>
<td>Yes</td>
<td>No</td>
<td>Yes (keys are immutable)</td>
<td>Yes (items are immutable)</td>
</tr>
<tr>
<td><strong>Iterable</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Comprehensions</strong></td>
<td>Yes (need to enclose in <code>tuple</code>)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Methods</strong></td>
<td><code>.append()</code>, <code>.extend()</code>, <code>.count()</code>, <code>.index()</code>, etc</td>
<td><code>.count()</code>, <code>.index()</code>, etc</td>
<td><code>.get()</code>, <code>.pop()</code>, etc</td>
<td><code>.add()</code>, <code>.remove()</code>, etc</td>
</tr>
</tbody>
</table>
# Python Data Structures at a Glance

<table>
<thead>
<tr>
<th></th>
<th>Lists</th>
<th>Tuples</th>
<th>Dictionaries</th>
<th>Sets</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Order</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Mutability</strong></td>
<td>Yes</td>
<td>No</td>
<td>Yes (keys are immutable)</td>
<td>Yes (items are immutable)</td>
</tr>
<tr>
<td><strong>Iterable</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Comprehensions**

- Yes
- Yes (need to enclose in `tuple`)
- Yes
- Yes

**Methods**

- `.append()`, `.extend()`, `.count()`, `.index()`, etc
- `.get()`, `.pop()`, `.add()`, `.remove()`, `.discard()`, etc

---

**Which to use when?**
Does Order Matter?

- Examples where order in data is important:
  - Ranked ballots
  - Queues
  - Words in a sentence
  - Tables/Matrices
- Tuples or lists?
  - Do we need to add/remove items dynamically?
    - If yes, use lists (they are mutable!)
    - If data stays same (no changes), use tuples (more space efficient)
    - Even though you can concatenate items to tuples, it is not efficient, as it requires “copying over all the data” and creating a new tuple
Unordered Collections

• When storing a collection of data with *no implicit ordering*:
  • Use **dictionaries** or **sets**
  • Dictionaries are more appropriate when there is a **key, value pair**
  • Better performance in general as compared to ordered structures

• Suppose we want to store student data in this course and quickly look up info for a given unix ID. Which data structure should we use?
  • Info may contain student name, class year, section, etc
  
• Can store a **dictionary of dictionaries** (just like lists of lists!)

```python
hpDict = { 'hp23': {'name': 'Harry James Potter', 'house':'Gryffindor', 'patronus': 'Stag'}, 'hg3': {'name': 'Hermione Jean Granger', 'house': 'Gryffindor', 'patronus': 'Otter'}, 'll4': {'name': 'Luna Lovegood', 'house': 'Ravenclaw', 'patronus': 'Hare'}}
```
Moving on...
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/graphics
- 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](http://matplotlib.org/examples)
- **pyplot documentation**: [http://matplotlib.org/api/pyplot_summary.html](http://matplotlib.org/api/pyplot_summary.html)
- **cool plots**: [https://matplotlib.org/gallery.html](https://matplotlib.org/gallery.html)
Plotting Basics: Plot function

```python
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)
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