CS106A

Juliette Woodrow

Housekeeping

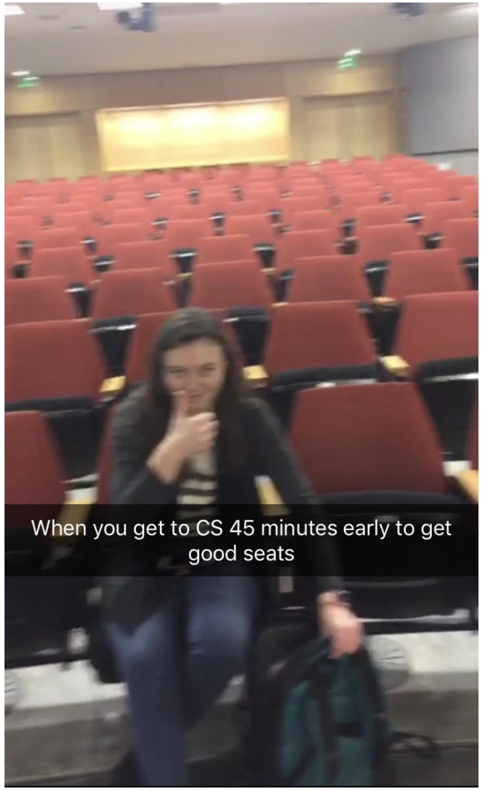
- Happy Monday! :)

- HW7 Due Wednesday

- Quiz 3 on Friday (finish your homework early to start studying :)
    - Review materials coming tomorrow
    - Section this week is focused on review
        - If you have a Friday section, maybe go to an earlier one
    - Material from HW6 and HW7 (no drawing on quiz 3)
    - Only covers up to last Friday’s lecture

A little bit about me...

A little bit about me...



A little bit about me...

Let’s Jump Right In

Guiding Questions

1. How can we make lists without map or for loops?

2. What tools do we have for developing and analyzing data?

3. What trends can we find in life expectancy, GDP, and population data over the last 215 years?

List Comprehensions

Problem: getting a list of squares

Problem: getting a list of squares

- Imagine you have a list of numbers, and you want a list of those same numbers squared

Problem: getting a list of squares

- Imagine you have a list of numbers, and you want a list of those same numbers squared

  [4, 6, 7, 8] → [16, 36, 49, 64]

Problem: getting a list of squares

- Imagine you have a list of numbers, and you want a list of those same numbers squared

  `[4, 6, 7, 8]` → `[16, 36, 49, 64]`

- How would you produce this output list?

Problem: getting a list of squares - Attempt #1

```
# [4, 6, 7, 8] → [16, 36, 49, 64]
```

Problem: getting a list of squares - Attempt #1

```
# [4, 6, 7, 8] → [16, 36, 49, 64]


def get_squared(num_lst):
    squares = []
    for num in num_lst:
        squares.append(num**2)
    return squares
```

Problem: getting a list of squares - Attempt #2

```
---

num_lst = [4, 6, 7, 8]
```

Problem: getting a list of squares - Attempt #2

```
---

num_lst = [4, 6, 7, 8]

list(map(lambda num: num**2, num_list))

# would also give us [16, 36, 49, 64]
```

Problem: getting a list of squares - Attempt #3

Problem: getting a list of squares - Attempt #3

```
---

num_lst = [4, 6, 7, 8]
```

Problem: getting a list of squares - Attempt #3

```
---

num_lst = [4, 6, 7, 8]

squared_lst = [num ** 2 for num in num_lst]
```

Problem: getting a list of squares - Attempt #3

```
___

num_lst = [4, 6, 7, 8]

squared_lst = [num ** 2 for num in num_lst]
```

this is a list comprehension!

List Comprehensions

```
[num ** 2 for num in num_lst]
```

List Comprehensions

```
[num ** 2 for num in num_lst]
```

- **Definition:** one way to make a new list based on the values of an existing list

List Comprehensions

```
[num ** 2 for num in num_lst]
```

- **Definition:** one way to make a new list based on the values of an existing list
- **Three Key Parts:**

List Comprehensions

```
[num ** 2 for num in num_lst]
```

$$\uparrow$$

expression

- **Definition:** one way to make a new list based on the values of an existing list
- **Three Key Parts:**
  - Expression

List Comprehensions

```
[num ** 2 for num in num_lst]
```

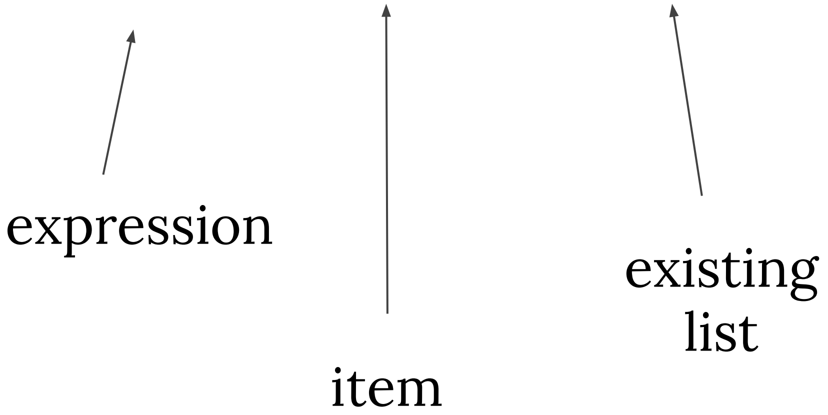
expression

item

- **Definition:** one way to make a new list based on the values of an existing list
- **Three Key Parts:**
  - Expression
  - Item

List Comprehensions

```
[num ** 2 for num in num_lst]
```



- **Definition:** one way to make a new list based on the values of an existing list
- **Three Key Parts:**
  - Expression
  - Item From Existing List
  - Existing List

Hey, we already know some of that syntax!

```
[num ** 2 for num in num_lst]
```

Hey, we already know some of that syntax!

```
[num ** 2 for num in num_lst]
```

- `[ ]` → **that makes it a list**

Hey, we already know some of that syntax!

`[num ** 2 for num in num_lst]`

- `[ ]` → that makes it a list
- `for num in  num_list` → that's just a for each loop

Hey, we already know some of that syntax!

```
[num ** 2 for num in num_lst]
```

- `[ ]` → that makes it a list
- **for num in num_list** → that's just a for each loop
- **n**2** → This is how we square a number

Let’s try it out!

Let’s try it out!

- You have a list of strings with random casing and you want a list of strings that are all lowercase

Let’s try it out!

- You have a list of strings with random casing and you want a list of strings that are all lowercase

  `[“Hi”, “mOm”, “aNd”, “DAD”]` → `[“hi”, “mom”, “and”, “dad”]`

Let’s try it out!

- You have a list of strings with random casing and you want a list of strings that are all lowercase

  [“Hi”, “mOm”, “aNd”, “DAD”] → [“hi”, “mom”, “and”, “dad”]

- How can we use a list comprehension to do this?

Problem: getting a list of lowercase strings

```
random_case = [“Hi”, “mOm”, “aNd”, “DAD”]
```

Problem: getting a list of lowercase strings

```
random_case = [“Hi”, “mOm”, “aNd”, “DAD”]

all_lower = [s.lower() for s in random_case]

print(all_lower)

# would print [“hi”, “mom”, “and”, “dad”]
```

Problem: converting temperature to fahrenheit

Problem: converting temperature to fahrenheit

- List of temperatures in degrees celsius

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]
```

Problem: converting temperature to fahrenheit

- List of temperatures in degrees celsius

  ```
  france_temps_c = [13, 14, 15, 16, 8, 9, 12]
  ```

- Want a list of temperatures in degrees fahrenheit

Problem: converting temperature to fahrenheit

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [ ]
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [ for t in france_temps_c]
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [t*(9/5) + 32 for t in france_temps_c]
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [t*(9/5) + 32 for t in france_temps_c]

print(france_temps_f)

# would print [55.4, 57.2, 59.0, 46.4, 48.2, 53.6, 46.4]
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [t*(9/5) + 32 for t in france_te

print(france_temps_f)

# would print [55.4, 57.2, 59.0, 46.4, 48.2, 53.6
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [t*(9/5) + 32 for t in france_te

print(france_temps_f)

# would print [55.4, 57.2, 59.0, 46.4, 48.2, 53.6
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]

℃(9/5) + 32  = ℉

france_temps_f = [t*(9/5) + 32 for t in france_temps_c]
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]
```

℃(9/5) + 32 = ℉

france_temps_f = [**t*(9/5) + 32** for t in france_temps_c]

- Can we decompose this?

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]
```

℃(9/5) + 32 = ℉

france_temps_f = [**t*(9/5) + 32** for t in france_temps_c]

- Can we decompose this? Yes !!

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]
```

℃(9/5) + 32 = ℉

france_temps_f = [**t*(9/5) + 32** for t in france_temps_c]

- Can we decompose this? Yes !!

```
def make_fahrenheit(c):
    return c * (9/5) + 32
```

Problem: converting temperature to fahrenheit

```
france_temps_c = [13, 14, 15, 16, 8, 9, 12]
```

℃(9/5) + 32 = ℉

france_temps_f = [**make_fahrenheit(t)** for t in france_temps_c]

- Can we decompose this? Yes !!

```
def make_fahrenheit(c):
    return c * (9/5) + 32
```

Conditions in List Comprehensions

- You can add a condition to a list comprehension for additional “filtering”

Conditions in List Comprehensions

- You can add a condition to a list comprehension for additional “filtering”

```
[expression for item in list if condition]
```

Conditions in List Comprehensions

- You can add a condition to a list comprehension for additional “filtering”

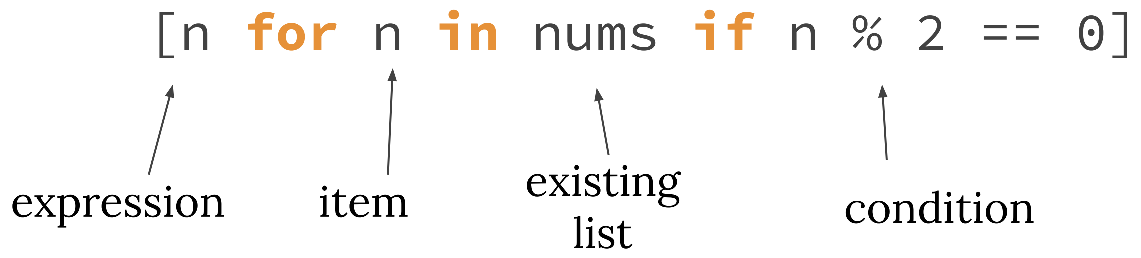
```
[expression for item in list if condition]
```

```
[n for n in nums if n % 2 == 0]
```

Conditions in List Comprehensions

- You can add a condition to a list comprehension for additional “filtering”

```
[expression for item in list if condition]
```



```
[n for n in nums if n % 2 == 0]
```

Let’s try it out

Let’s try it out

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

Let’s try it out

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

Let’s try it out

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]
```

Let’s try it out

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]

# y_at_end_kids is [“henry”, “audrey”, “bailey”]
```

Let’s try it out

Note: a list comprehension makes a *new* list and does not modify the original one

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]

# y_at_end_kids is [“henry”, “audrey”, “bailey”]
```

Let’s try it out

Note: a list comprehension makes a *new* list and does not modify the original one

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]

# y_at_end_kids is [“henry”, “audrey”, “bailey”]
```

- Want only the names that start with “b”

Let’s try it out

Note: a list comprehension makes a *new* list and does not modify the original one

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]

# y_at_end_kids is [“henry”, “audrey”, “bailey”]
```

- Want only the names that start with “b”

```
b_at_front_kids = [name for name in kids if name[0] == ‘b’]
```

Let’s try it out

Note: a list comprehension makes a *new* list and does not modify the original one

```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]
# y_at_end_kids is [“henry”, “audrey”, “bailey”]
```

- Want only the names that start with “b”

```
b_at_front_kids = [name for name in kids if name[0] == ‘b’]
#b_at_front_kids is
```

Let’s try it out

Note: a list comprehension makes a *new* list and does not modify the original one

```
kids = [“jona                 ry”, “juliette”, “audrey”, “bailey”]
 ●  Want on                  d in “y”
y_at_end_ki                  in kids if name[-1] == ‘y’]
# y_at_end_                 audrey”, “bailey”]
 ●  Want on                  art with “b”
b_at_front_                 me in kids if name[0] == ‘b’]
#b_at_front
```

Let’s try it out

Note: a list comprehension makes a *new* list and does not modify the original one

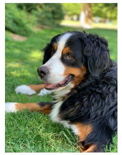
```
kids = [“jonathan”, “isabelle”, “henry”, “juliette”, “audrey”, “bailey”]
```

- Want only the names that end in “y”

```
y_at_end_kids = [name for name in kids if name[-1] == ‘y’]
# y_at_end_kids is [“henry”, “audrey”, “bailey”]
```

- Want only the names that start with “b”

```
b_at_front_kids = [name for name in kids if name[0] == ‘b’]
#b_at_front_kids is [“bailey”]
```



Why List Comprehensions?

- They make me feel cool 😎
- They are more concise
- They are *Pythonic*

Why List Comprehensions?

- They make me feel cool 😎
- They are more concise
- They are *Pythonic*

What does it mean to by *Pythonic*?

What tools do we have to develop and analyze data?

Let’s Analyze Some Data

Dataset Source

Let’s Analyze Some Data

- Found this cool dataset with life expectancy, GDP, and populations of countries over the last 215 years

Dataset Source

Let’s Analyze Some Data

- Found this cool dataset with life expectancy, GDP, and populations of countries over the last 215 years

- Three files: life.csv, gdp.csv, pop.csv

Dataset Source

Let’s Analyze Some Data

- Found this cool dataset with life expectancy, GDP, and populations of countries over the last 215 years

- Three files: life.csv, gdp.csv, pop.csv

- Each line in the file looks like this:
  - country_name,stat_year1,stat_year2,stat_year3, … ,stat_year215
  - Where year 1 is 1800

Dataset Source

Let’s check out the code

Jupyter Notebook

Jupyter Notebook

Jupyter Notebook

- Interactive “notebook” where you can run parts of your code at a time

Jupyter Notebook

- Interactive “notebook” where you can run parts of your code at a time
  - Can develop code step by step
  - Great for data analysis

Jupyter Notebook

- Interactive “notebook” where you can run parts of your code at a time
  - Can develop code step by step
  - Great for data analysis

- Built on top of regular python

Jupyter Notebook

- Interactive “notebook” where you can run parts of your code at a time
  - Can develop code step by step
  - Great for data analysis

- Built on top of regular python

- Kind of like a playground for you to work in
- Supplemental to Pycharm

Jupyter Notebook

- Interactive “notebook” where you can run parts of your code at a time
  - Can develop code step by step
  - Great for data analysis

- Built on top of regular python

- Kind of like a playground for you to work in
- Supplemental to Pycharm

- Good for collaboration !
  - In your CS life outside of this class, you will likely collaborate on the code that you write

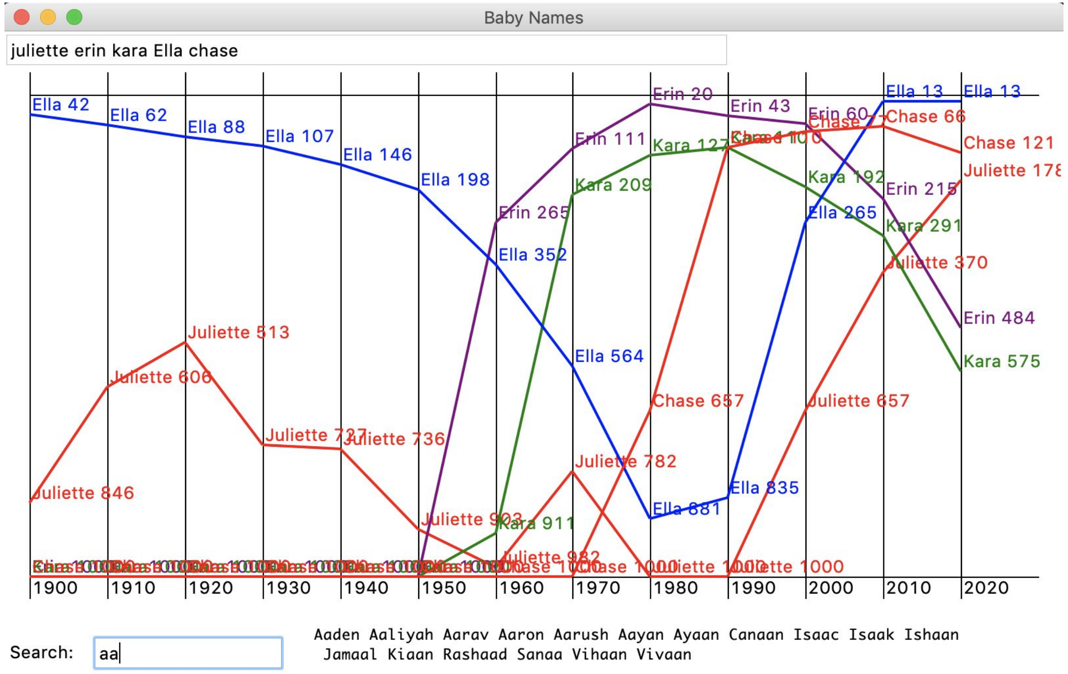
Jupyter Notebook

- Jupyter Notebook and Matplotlib Set Up

```
py -m pip install jupyter # (use python3 instead of py on Mac)

py -m pip install matplotlib # (use python3 instead of py on Mac)
```

How Can We Visualize Data?

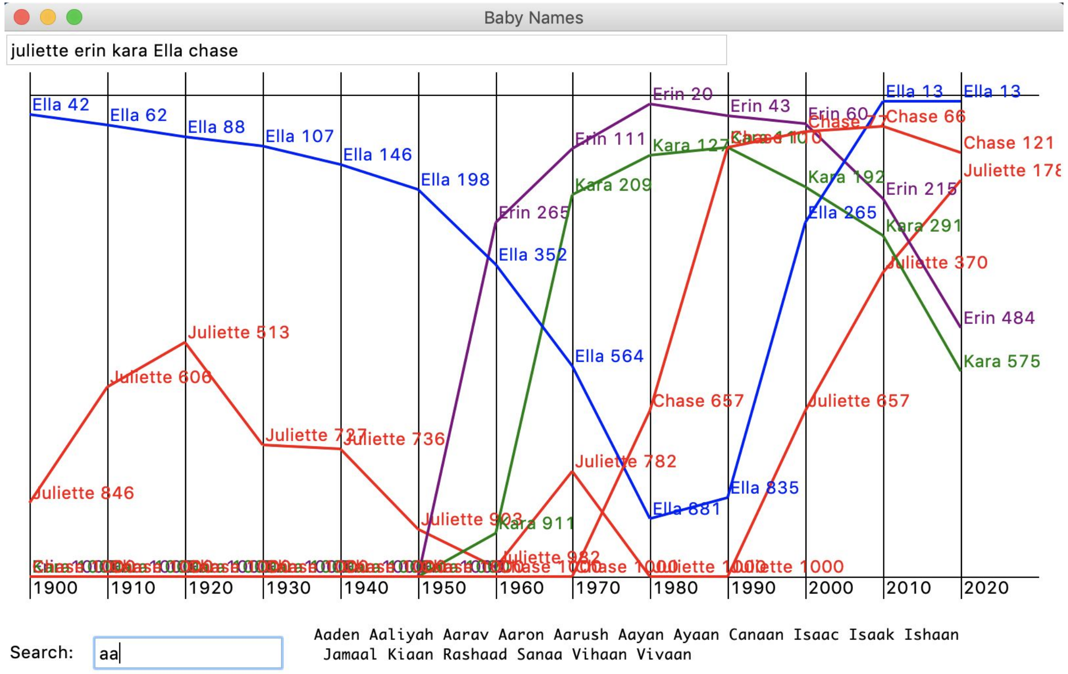


Baby Names

juliette erin kara Ella chase

Search: aa

```
Aaden Aaliyah Aarav Aaron Aarush Aayan Ayaan Canaan Isaac Isaak Ishaan
 Jamaal Kiaan Rashaad Sanaa Vihaan Vivaan
```


Baby Names

juliette erin kara Ella chase

Search: aa

Aaden Aaliyah Aarav Aaron Aarush Aayan Ayaan Canaan Isaac Isaak Ishaan Jamaal Kiaan Rashaad Sanaa Vihaan Vivaan

You are all already experts on this!
But...

there is another way

Using Matplotlib

Using Matplotlib

- A library to create plots
  - Other people felt your pain and they created a library to help us all make graphs in python

Using Matplotlib

- A library to create plots
  - Other people felt your pain and they created a library to help us all make graphs in python

- To install
  - `$ py -m pip install matplotlib #(use python3 instead of py on Mac)`

Using Matplotlib

Using Matplotlib

```
import matplotlib.pyplot as plt
```

Using Matplotlib

```
import matplotlib.pyplot as plt

# x = list of x vals;     y = list of y vals

plt.plot(x, y)   # line plot
```

Using Matplotlib

```
import matplotlib.pyplot as plt

# x = list of x vals;     y = list of y vals

plt.plot(x, y)   # line plot

plt.scatter(x, y)   # scatter plot
```

Using Matplotlib

```
import matplotlib.pyplot as plt

# x = list of x vals;     y = list of y vals

plt.plot(x, y)   # line plot

plt.scatter(x, y)   # scatter plot

plt.bar(x, y)   # bar plot

plt.title(text)   #adds a title to the plot

plt.show()    #displays the plot
```

Using Matplotlib

- There are many more features !!
  - Read more about Matplotlib here

  - This is a useful Matplotlib tutorial tutorial

Let’s try it out

Jupyter Summary

Jupyter Summary

- py -m pip install Jupyter
- Open homework folder in Pycharm like normal
- From terminal in Pycharm type: jupyter notebook
    - This command will open the notebook in your browser
    - Navigate to the .ipynb file that you want to work in

- To run a cell in Jupyter hit: shift+enter

- Can rerun smaller amounts of code at a time to answer questions about datasets

Matplotlib Summary

Matplotib Summary

- Matplotlib is a massive module
- We looked at the pyplot interface within matplotlib today
    - More documentation on matplotlib.pyplot here

- What you need to know for assignment 8:
    - import matplotlib.pyplot as plt #to use in Jupyter notebook
    - # make a list of x and y values that you want to plot
    - plt.bar(x_vals, y_vals, color=”tab:color_name”)
    - How to add titles and tables
        - plt.title(“Cool title”)
        - plt.xlabel(“Awesome x label”)
        - plt.ylabel(“Awesome y label”