Announcements

- Homework
    - Homework 8 and 9 due tonight!
    - Only 1 more homework!
- Starting in on visualization today (Ch 11)
- Will be getting information to you next week concerning projects
- Polling: `rembold-class.ddns.net`

Review Question

```python
class MyClass:
    varA = 3
    varB = True

    def __init__(self):
        self.v = self.varA
        if self.varB:
            self.varA += 1

A = MyClass()
B = MyClass()
MyClass.varB = False
C = MyClass()
print(MyClass.varA)
```

Suppose the code to the left was written and executed. What would be the output of the printed statement?

A) 3
B) 5
C) 6
D) None of the above

Solution: 3

Visualization

- Interfaces mostly text so far:
  - Written to screen
  - Written to a file
  - Input from screen or from file
- Want to extend how we can communicate results or interface with our programs
  - Data representation and plotting
  - More complex graphical output/input

Reminder: Using Modules

- We already know how to import and write our own modules
- Many topics going forward will focus heavily on extending Python through the use of a common or core module
    - Modules are usually written in classes!
    - Create new objects with special attributes and properties
    - Interact with those objects through specific methods
    - Read the documentation for a module or a method to learn what is available and how it should be used

Matplotlib

- *The* fundamental plotting and data visualization package for Python
  - Old! Released 16 years ago!
- Has two main ways to interface with the objects:
  - A state-based interface based on MATLAB
  - An object-oriented interface
- Even using the OO interface, the state-based library has some useful bits, so that is still what we will import:

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

Matplotlib

- *The* fundamental plotting and data visualization package for Python
    - Old! Released 16 years ago!
- Has two main ways to interface with the objects:
    - A state-based interface based on MATLAB
    - An object-oriented interface
- Even using the OO interface, the state-based library has some useful bits, so that is still what we will import:

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

The Objects

- Two primary objects you'll use in Matplotlib:
  - A figure
    - The overall image that is created

```
fig = plt.figure()
```

  - An axis
    - A coordinate system into which data can be displayed
    - Is inserted into a figure
    - You can have multiple axes in a single figure if desired

```
ax = fig.add_axes()
# or, more commonly
ax = fig.add_subplot(111)
```

Actually Plotting

- You can add a plot to a desired axes instance
  - Use the `.plot()` method
- Plots are 2D
  - Need a sequence of y coordinates
  - Give a sequence of x coordinates or indices will be default
  - x coordinates are given first if they exist
- Style of the plot can be controlled with special format string as second/third parameter.
- Additional properties of a plot can be controlled with extra keyword parameters.

Plot Customization

- Can determine how a plot generally looks with a format string:
  - Takes the form of

    ```
    '[marker][line][color]'
    ```

    - Common markers: `. o ^ * s D`
    - Common lines: `- -- :`
    - Common colors: `b g r c k`
- Can also add keyword arguments:
  - Change color: `color = 'green'`
  - Charge marker size: `markersize = 20`
  - Charge opacity: `alpha = 0.4`
  - Add label for legend: `label = 'Best plot'`

Extra Figure and Axes Features

- Clear communication is important with visualization
- Should always include meaningful and descriptive axes labels and figure titles.
  - Axes labels controlled with `.set_xlabel()` and `.set_ylabel()`
  - Axes title controlled with `.set_title`
  - Figure title controlled with `.suptitle`
- Adjust tick spacing or labels:
  - Where ticks appear: `.set_xticks()` or `.set_yticks()`
  - What labels they have: `.set_xticklabels()` or `.set_yticklabels()`