EN.540.635 "Software Carpentry"

Weekly Challenge 5 - Object-Oriented Plotting

The purpose of this assignment is to become more familiar with making user-defined classes in Python and also using some built-in functionalities from Matplotlib to make figures. We've seen lambda functions and how they can be used in place of the typical function definitions we are used to seeing. The most useful application of lambda functions is for representing math functions. Let's say that we want to be able to combine functions (add, subtract, multiply, and divide them) easily and also plot them. While there is a way to do these things procedurally, we can make use of classes to simplify these tasks even further. In this assignment, we will be writing two specific classes to help us with this, which are outlined below.

The Function Class:

- This class will act as a wrapper for a generic lambda function. We will have to define the __init__ and __call__ methods appropriately.
- We can define some default methods to appropriately handle combining functions together in different ways. Specifically, we want this class to handle the addition, subtraction, multiplication, and division of two Function classes.
- This class should also be able to handle float values appropriately.

The Plotter Class:

- This class will act as a wrapper for Matplotlib's plotting function. When we initialize a Plotter object, we want to be able to specify the domain of the function and the step size. We also want this class to be able to hold Function objects.
- We want to write user-defined methods for adding Function objects to our Plotter class and plotting all of them.

Let's say we have the following three functions:

$$f_1(x) = x^2$$

$$f_2(x) = x + 3$$

$$f_3(x) = \sin(x)$$

We want to combine these three functions together in the following way to create a fourth function:

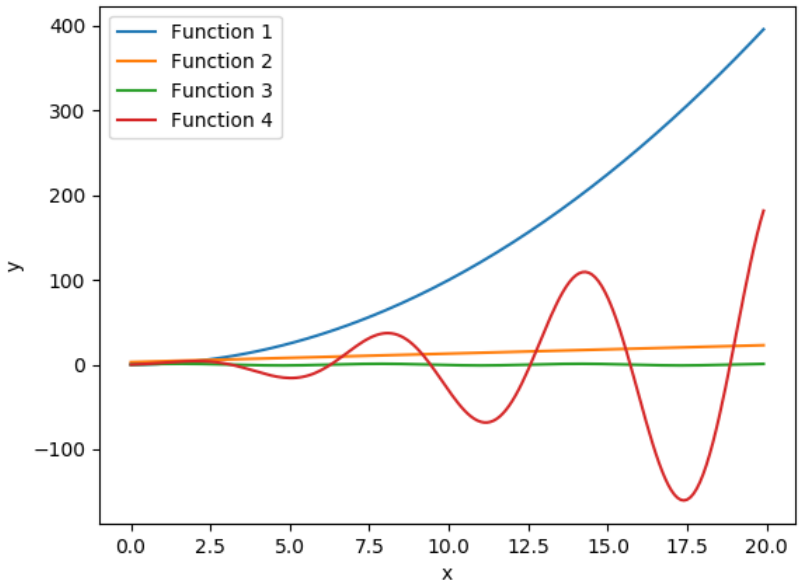
$$f_4(x) = \frac{(f_1+f_2)f_3}{2} = \frac{(x^2+x+3)\,\sin(x)}{2}$$

If we've defined these two classes appropriately, the following main block of code should run. The comments here are specifying exactly what should be input to the classes.

```
if __name__ == "__main__":

    # For each Function object, we simply input the lambda function to it.
    # Note: be sure to include "import numpy as np" at the beginning of your
    # Python script so that we can make use of numpy's sine function.
    f1 = Function(lambda x: x ** 2)
    f2 = Function(lambda x: x + 3)
    f3 = Function(lambda x: np.sin(x))

    # Combine our first 3 Function objects together to create a new fourth
    # Function object.
    f4 = (f1 + f2) * f3 / 2.0
```

```
# Create a new Plotter object, where our domain ranges from 0 to 20 and we
# have a step size of 0.1.
plot = Plotter(0, 20, 0.1)

# Add all the functions to the Plotter object. The first argument specifies
# the function's label and the second argument is the actual Function
# object. Keep in mind that you must write the functions add_func and plot.
plot.add_func("Function 1", f1)
plot.add_func("Function 2", f2)
plot.add_func("Function 3", f3)
plot.add_func("Function 4", f4)
plot.plot()
```

After this code has been run, we should be able to get a figure that looks like this:



Please upload a Python script to the appropriate assignmnent on Blackboard that contains the following components:

1. The Function class with all the appropriate methods as described previously.
2. The Plotter class with all the appropriate methods as described previously.
3. The main block of code shown above. Note: If you copy and paste this text directly from this PDF, there will be weird spaces that appear in the pasted text. Make sure you edit the code to match what is written.

Also, make sure you keep the following things in mind:

1. Make sure you import all the necessary modules needed at the beginning of the Python script (import numpy as np, import matplotlib.pyplot as plt).
2. All classes and user-defined functions should have appropriate docstrings.
3. Include appropriate comments if needed. More details about this are in the slides for Lecture 6.
4. You are allowed to write extra functions (either inside or outside of your class definitions) to help you accomplish the requirements of the assignment.