UCF Physics PHZ 3150: Introduction to Numerical Computing
Fall 2021 Homework 9
Due October 28 2021

Goals: practice making functions, coding any problem you have and making your own classes.
Reading: TP2 chapters 15, 16, 17, 18

Problem 1 (**5 points**). Make a new folder named `hw9_<yourname>` under you `handin` folder. As always, your log is part of your homework. After each problem number, give your answer and the names of any files you are handing in for each problem. If you made a HW9 entry in your log in a prior session and want to change it, just copy it to the current (last) session, and edit there. We will grade the last entry only. All text related to one assignment should be in one entry, with the problems done in order.

Problem 2 (**30 points total**). You are in the committee for organizing your town's triathlon and you need to have an estimate for how long the event will take, and who might win it.

a) (5 points) Create a numpy array `sports` with the average distance per sport (swim, bike, run) : 1500, 40000, 10000.
'Mapping' the names of athletes to an ID number (e.g, Mary = 1, John = 2 etc) create a numpy array `athletes_times` that contains information about each participant's average speed per sport (so you will have a (6,4) array):

Participant	ID	m/s for swimming	m/s for biking	m/s for running
Mary	1	1.204	6.6	0.32
John	2	1.212	6.5	0.36
Peter	3	1.14	6.6	0.38
Mafalda	4	1.12	6.2	0.38
Paul	5	1.24	5.6	0.4
Lionel	6	1.201	6.0	0.3

b) (10 points) Create a function `triathlon_time(sports,athletes_times)` that gets as input your two arrays and outputs an informative message about which participant will finish first and their expected time in hours, and which one will finish last and their expected time in hours. Remember to use an appropriate docstring.

c) (4 points) Change the list sports into a dictionary `sports` with keywords the sports, and values the total distance for each sport. Change `athletes_times`

into a dictionary that has as keywords the names of participating athletes. For each participant (i.e., for each key) create a list with their average speed per sport (i.e., the appropriate dictionary values).

d) (8 points) Create function `triathlon_time_dict(sports,athletes_times)` that get as input your two dictionaries and like `triathlon_time(sports,athletes)` outputs an informative message about which participant will finish first -- and their expected time in hours -- and which one will finish last and their expected time in hours.

   Tip 1: to simplify the problem you can gather the total distances the athletes need to cover in a list or array in your `triathlon_time_dict`.
   Tip 2: Scan over your athletes dictionary, and using their average speed per sport calculate the total time each one needs to finish the race. Find the minimum and maximum of these times and use that for your print() statements.

e) (3 points) Call your functions (both `triathlon_time` and `triathlon_time_dict`) and let me know who finishes first and who last, and what their times will be.

Problem 3 (**20 points**) Do your homework(!) In Python.

a) (5 points) Choose any problem from a recent homework assignment or other assessment in another class that involves something you need to calculate or derive and that you can plot. State the class, section, assignment, and item number. State (or scan) the problem. State or scan your solution as given in that class (corrected by you).

b) (15 points) Now, code your solution and make and save one or more plots. Your code must use two or more of: control flow structures (conditional, loops, function calls), when possible avoid loops and use NumPy array math (you'll get **bonus 5 points** for actively doing this!), broadcasting, string formatting, reading or writing data to/from a file, masks. However, everything you write should be natural and belong in the code.

Problem 4 (**15 points**) Holidays are coming and you need to find out how much wrapping paper you will need and how many presents you can fit in every box you want to put under the tree! Create a class `holiday_frenzy` that uses the dimensions `a,b,c` of a box to calculate the box's surface area and volume. Assume that your dimensions are in inches. The class should also know how much wrapping paper you

have (in inches^2) and how many presents you want to fit in your box. Use a proper initiation function.

Your class should contain:

a) An initiation function
b) a function that returns the surface area of the box (e.g., `surface_area`),
c) a function that returns the volume of the box (e.g., `volume`),
d) a function that checks if you have enough wrapping paper (`enough_paper`)
e) a function that checks if your box can fit all the presents (`fits;` assuming that the average volume of a gift is 25 inches^3) and
f) a function (`print_gift`) that combines all these information and lets you know (with print() statements) if you have enough wrapping paper to wrap your box, if your gifts fit in the box, or if nothing is right.

   Test it `for (a, b, c, number_of_gifts, wrap_paper_available) = (10,10,10,15,800)` ; `(10,10,10,55,1000)` and `(100,20,10,55,300)`.

Problem 5 (**10 points**). Prepare and submit your homework. Write what you will do to make and submit the zip file into your log. Don't forget to also commit your finalized log and push it to GitHub. When satisfied, close the log, copy it to your homework directory, and run the commands to make and submit the zip file. Turn the file in on WebCourses.