


Assignment 1


[New Attempt](#)

Due Sep 23 by 11:59pm **Points** 100 **Submitting** a website url

Create a private repository on github.com (<https://github.com>) for this course, share it with **ramcdougal** and **LejiaHu776**, and submit the URL on canvas. Include in your repository a folder for this homework with separate Python scripts (.py files) or notebooks (.ipynb files) for each exercise (these must clearly generate your answers to the various questions) and *a single readme file in pdf or markdown* that contains any instructions necessary for running your scripts, answers to the specific questions asked in the exercises, and all the code as an appendix. Thus the readme file contains everything needed to understand your work and the rest of the repo has runnable code so we can test it, etc.

Feel free to discuss basic Python questions with your classmates, the TF, and the professor in real life or on the relevant [discussion thread](#), but unless you're speaking with the TF or the professor, neither you nor the person you're talking to should be looking at your homework solutions while doing so.

You are not being graded on programming style for this assignment. Nevertheless, please try to follow the "[good enough practices](https://doi.org/10.1371/journal.pcbi.1005510)  (<https://doi.org/10.1371/journal.pcbi.1005510>)", including: meaningful variable and function names, use of functions, use of comments.

If there are any questions about the policies for homework, how to write the readme, what any of the questions mean, how to submit it, etc, please ask as early as possible. *If you find yourself stuck on a problem for an extended time, please ask for help. For extra hints, see [this cheatsheet](https://yale.instructure.com/courses/81264/files/6637528/download?download_frd=1) * (https://yale.instructure.com/courses/81264/files/6637528/download?download_frd=1) (in Jupyter notebook form) courtesy of Max Wibert.

Your repository must be shared and work must be complete by 11:59pm on September 23rd.

Exercise 1. (25 points).

Write a function `temp_tester` that takes a definition of normal body temperature and returns a function that returns `True` if its argument is within 1 degree of normal temperature (i.e. the value is a healthy temperature), and `False` if not (12 points). Test your code with the following (include a copy of your tests in your [GitHub \(https://github.com\)](https://github.com) repository) (13 points):

```
human_tester = temp_tester(37)
chicken_tester = temp_tester(41.1)

chicken_tester(42) # True -- i.e. not a fever for a chicken
human_tester(42)   # False -- this would be a severe fever for a human
chicken_tester(43) # False
human_tester(35)   # False -- too low
human_tester(98.6) # False -- normal in degrees F but our reference temp was in degrees C
```

Important: this question is not really about chickens. *Do not hard-code special cases for chickens and humans.* This should also work for whatever temperature we consider normal.

In terms of our "Good enough practices", having a generic function that makes other functions as needed is a function (2b) that itself eliminates duplication (2c) by generating these other functions. We're not testing libraries here (2e), but testing our code is obviously important too.

Exercise 2. (25 points)

Download the sqlite3 database from [hw1-population.db](https://yale.instructure.com/courses/81264/files/6635791/download?download_frd=1) ↓

(https://yale.instructure.com/courses/81264/files/6635791/download?download_frd=1)

(Note: this is a link to a page where you can download the database; it is not the database.)

Use the following code to load it into Python as a pandas DataFrame:

```
import pandas as pd
import sqlite3
with sqlite3.connect("hw1-population.db") as db:
    data = pd.read_sql_query("SELECT * FROM population", db)
```

Examine data. What columns does it have? (2 points) How many rows (think: people) does it have? (2 points)

Examine the distribution of the ages in the dataset. In particular, be sure to have your code report the mean, standard deviation, minimum, maximum. (2 points) Plot a histogram of the distribution with an

appropriate number of bins for the size of the dataset (describe in your readme the role of the number of bins). (3 points) Comment on any outliers or patterns you notice in the distribution of ages. (1 point)

Repeat the above for the distribution of weights. (3 points)

Make a scatterplot of the weights vs the ages. (3 points) Describe the general relationship between the two variables (3 points). You should notice at least one outlier that does not follow the general relationship. What is the name of the person? (3 points) Be sure to explain your process for identifying the person whose values don't follow the usual relationship in the readme. (3 points)

Exercise 3. (25 points).

Download historical data for COVID-19 cases by state from The New York Times's GitHub at <https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-states.csv> (<https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-states.csv>). (The full repository including licensing terms is at github.com/nytimes/covid-19-data (<https://github.com/nytimes/covid-19-data>)). As this is an ongoing public health crisis, include in your readme the date you downloaded the data (2 points). Since you are using data from an external source, be sure to credit The New York Times as your data source in your readme as well (2 points).

Load the data into Python in any way you wish. Here's one way that gives you a DataFrame assuming you've put the data in your working directory:

```
import pandas as pd
data = pd.read_csv("us-states.csv")
```

Make a function that takes a list of state names and plots their new cases vs date using overlaid line graphs, one for each selected state. (Note: the data file shows running totals, so you'll have to process it to get new case counts.) Be sure to provide a way to tell which line corresponds to what state (one possibility: using colors and a legend). If your approach has any specific limitations, explain them in your readme. (4 points)

Test the above function and provide examples of it in use. (4 points)

Make a function that takes the name of a state and returns the date of its highest number of new cases. (4 points)

Make a function that takes the names of two states and reports which one had its highest number of **daily new** cases first and how many days separate that one's peak from the other one's peak. (5

points) (Edit: 2022-09-14: clarification that we're talking about the peak of daily new cases, not of the total number of cases.)

Test the above function and provide examples of it in use. (4 points)


(Hint: [here's an example](#)

https://colab.research.google.com/drive/1jsXkf9zfnxXLi_kqMRcRd8sBOAlsrlsZ?usp=sharing) an example of using [dateutil.parser](https://dateutil.readthedocs.io/en/stable/parser.html) (<https://dateutil.readthedocs.io/en/stable/parser.html>) to find the number of days between two dates. This is not part of the baseline Python but it is a common library that is distributed with e.g. Anaconda. Pandas can also work with dates directly.)

Exercise 4 (25 points).

Download the MeSH data desc2022.xml from

https://nlmpubs.nlm.nih.gov/projects/mesh/MESH_FILES/xmlmesh/ 

(https://nlmpubs.nlm.nih.gov/projects/mesh/MESH_FILES/xmlmesh/) (A guide to MeSH XML is available at: <https://www.nlm.nih.gov/mesh/xmlmesh.html> ) (<https://www.nlm.nih.gov/mesh/xmlmesh.html>) You'll probably want to look at a snippet of the file to get a sense of how it's written.

Write Python code that reads the XML and reports:

- the DescriptorName associated with DescriptorUI D007154 (the text of the name is nested inside a String tag) (5 points)
- the DescriptorUI (MeSH Unique ID) associated with DescriptorName "Nervous System Diseases" (5 points)
- the DescriptorNames of items in the MeSH hierarchy that are **descendants** of both "Nervous System Diseases" and D007154. (That is, each item is a subtype of both, as defined by its TreeNumber(s).) (5 points)

(Edit: 2022-09-21: replaced "children" with "descendants"... i.e. we're not looking at just the immediate children. The explanation remains unchanged.)

Explain briefly in terms of biology/medicine what the above search has found. (5 points)

Do these tasks using functions (e.g. write a generic function that returns DescriptorName given a DescriptorUI) instead of writing single use code. (5 points)