Homework 4: Functions, Histograms, and Groups

Reading:

- Visualizing Numerical Distributions (https://www.inferentialthinking.com/chapters/07/2/visualizing-numerical-distributions.html)
- Functions and Tables (https://www.inferentialthinking.com/chapters/08/functions-and-tables.html)

Deadline:

This assignment is due Thursday, March 4 at 11:59 P.M. Late work will not be accepted as per the course policies (see the Syllabus and Course policies on Canvas (https://canvas.yale.edu).

Directly sharing answers is not okay, but discussing problems with the course staff or with other students is encouraged. Refer to the policies page to learn more about how to learn cooperatively.

You should start early so that you have time to get help if you're stuck. The drop-in office hours schedule can be found on Canvas (https://canvas.yale.edu). You can also post questions or start discussions on Ed Discussion (https://edstem.org/us/courses/3558/discussion/).

Submission:

Submit your assignment as a .pdf on Gradescope. You can access Gradescope through Canvas on the left-side of the class home page. The problems in each homework assignment are numbered. NOTE: When submitting on Gradescope, please select the correct pages of your pdf that correspond to each problem. This will allow graders to find your complete solution to each problem.

To produce the .pdf, please do the following in order to preserve the cell structure of the notebook:

1. Go to "File" at the top-left of your Jupyter Notebook
2. Under "Download as", select "HTML (.html)"
3. After the .html has downloaded, open it and then select "File" and "Print" (note you will not actually be printing)
4. From the print window, select the option to save as a .pdf

Note:

Please complete this notebook by filling in the cells provided. Before you begin, execute the following cell to load the provided tests. Each time you start your server, you will need to execute this cell again to load the tests.

In [ ]:

```python
# Don't change this cell; just run it.

import numpy as np
from datascience import *

# These lines do some fancy plotting magic.\n",
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plots
plots.style.use('fivethirtyeight')
```

1. Working with Text using Functions

The following table contains the words from four chapters of Charles Dickens' *A Tale of Two Cities* (http://www.gutenberg.org/cache/epub/98/pg98.txt). We're going to compute some simple facts about each chapter. Since we're performing the same computation on each chapter, it's best to encapsulate each computational procedure in a function, and then call the function several times. Run the cell to get a table with one column.

In [ ]:

```python
# Just run this cell to load the data.
tale_chapters = Table.read_table("tale.csv")
tale_chapters
```

Question 1.1. Write a function called `word_count` that takes a single argument, the text of a single chapter, and returns the number of words in that chapter. Assume that words are separated from each other by spaces.

Hint: Try the string method `split` (https://docs.python.org/3/library/stdtypes.html#str.split) and the function `len` (https://docs.python.org/3/library/functions.html#len).

In [ ]:

```python
...

# Keep the line below to check the output of the function you write above
word_count(tale_chapters.column("Chapter text").item(0))
```

Question 1.2. Create an array called `chapter_lengths` which contains the length of each chapter in `tale_chapters`.

Hint: Consider using `apply` along with the function you have defined in the previous question.

In [ ]:

```python
chapter_lengths = ...
chapter_lengths
```

Question 1.3. Write a function called `character_count`. It should take a string as its argument and return the number of characters in that string that aren't spaces (" "), periods ("."), exclamation marks ("!"), or question marks ("?"). Remember that `tale_chapters` is a table, and that the function takes in only the text of one chapter as input.

Note: Unlike Question 1.1, we have not included a line at the bottom of the next cell to run your function and produce the output. Make sure you include that line yourself after you write your function so we can see your output.

Hint: Try using the string method `replace` several times to remove the characters we don't want to count.

In [ ]:

```
...
```

Question 1.4. Write a function called `chapter_number`. It should take a single argument, the text of a chapter from our dataset, and return the number of that chapter, as a Roman numeral. (For example, it should return the string "I" for the first chapter and "II" for the second.) If the argument doesn't have a chapter number in the same place as the chapters in our dataset, `chapter_number` can return whatever you like.

To help you with this, we've included a function called `text_before`. Its documentation describes what it does.

Note: After you write the function, make sure to include a line of code at the bottom of the next cell that produces the output we want according to the instructions in this question.

```
In [ ]: def text_before(full_text, pattern):
            """Finds all the text that occurs in full_text before the specified
         pattern.

            Parameters
            ----------
            full_text : str
                The text we want to search within.
            pattern : str
                The thing we want to search for.

            Returns
            -------
            str
                All the text that occurs in full_text before pattern.  If patter
        n
                doesn't appear anywhere, all of full_text is returned.

            Examples
            --------

            >>> text_before("The rain in Spain falls mainly on the plain.", "Spa
        in")
            'The rain in '
            >>> text_before("The rain in Spain falls mainly on the plain.", "ai
        n")
            'The r'
            >>> text_before("The rain in Spain falls mainly on the plain.", "Por
        tugal")
            'The rain in Spain falls mainly on the plain.'
            """
            return np.array(full_text.split(pattern)).item(0)

        def chapter_number(chapter_text):
            ...
```

2. Uber

Below we load tables containing 200,000 weekday Uber rides in the Manila, Philippines, and Boston, Massachusetts metropolitan areas from the Uber Movement (https://movement.uber.com) project. The `sourceid` and `dstid` columns contain codes corresponding to start and end locations of each ride. The `hod` column contains codes corresponding to the hour of the day the ride took place. The `ride time` table contains the length of the ride, in minutes.

In [ ]:

```
# Just run this cell to load the data.
boston = Table.read_table("boston.csv")
manila = Table.read_table("manila.csv")
print("Boston Table")
boston.show(4)
print("Manila Table")
manila.show(4)
```

Question 2.1. Produce histograms of all ride times in Boston and in Manila, using the given bins. Please put the code for both of them in the following cell, and put the ride times for Boston first.

In [ ]:

```
bins = np.arange(0, 120, 5)
...
...
```

Question 2.2. Set the two variables below to estimates of what percentage of rides are less than 10 minutes in Boston and Manila. Find your estimates by visually assessing the histograms. Your solution should consist of only mathematical operations and numbers.

In [ ]:

```
boston_under_10 = ...
manila_under_10 = ...
```

Question 2.3. Comment on the main difference between the two histograms. What might be causing this?

Your Answer Here:

The following two questions are optional!

Please do make an attempt at them, but they will not be incorporated into the final grading of this homework.

Optional Question 2.4. The `hod` column in each table represents the hour of the day during which the Uber was called. 0 corresponds to 12-1 AM, 1 to 1-2 AM, 13 to 1-2 PM, etc. Write a function which takes in a table like `boston` or `manila`, and an `hod` number between 0 and 23, and displays a histogram of ride lengths from that hour in that city. Use the same bins as before.

In [ ]:

```
def hist_for_time(tbl, hod):
    bins = np.arange(0, 120, 5)
    ...

#DO NOT DELETE THIS LINE!
hist_for_time(boston, 12)
```

Optional Question 2.5. Which city has a larger difference between Uber ride times at 10 AM vs. 10 PM? In other words, which is larger: the difference between 10 AM and 10 PM Uber ride times in Manila or the difference between 10 AM and 10 PM uber ride times in Boston. Use the function you just created to answer this question. You do not need to calculate an actual difference.

Assign `larger_diff` to the number 1 if the answer is Manila, and 2 if the answer is Boston.

In [ ]:

```
larger_diff = ...
```

3. Faculty salaries

In the next cell, we load a dataset created by the Daily Cal (http://projects.dailycal.org/paychecker/) which contains Berkeley faculty, their departments, their positions, and their gross salaries in 2015. (Unfortunately this information is not publicly available for Yale!)

In [ ]:

```
# Just run this cell to load the data.
raw_profs = Table.read_table("faculty.csv").where("year", are.equal_to(2
015)).drop("year", "title")
profs = raw_profs.relabeled("title_category", "position")
profs
```

We want to use this table to generate arrays with the names of each professor in each department.

Question 3.1 Set `prof_names` to a table with two columns. The first column should be called "department" and have the name of every department once, and the second column should be called "faculty" and contain an *array* of the names of all faculty members in that department.

Hint: Think about how `group` works: it collects values into an array and then applies a function to that array. We have defined two functions below for you, and you will need to use one of them in your call to `group` .

In [ ]:

```
# Pick between the two functions defined below
def identity(array):
    return array

def first(array):
    return array.item(0)
```

In [ ]:

```
prof_names = ...
prof_names
```

Question 3.2 At the moment, the `name` column is sorted by last name. Would the arrays you generated in the previous part be the same if we had sorted by first name instead before generating them? Two arrays are the **same** if they contain the same number of elements and the elements located at corresponding indexes in the two arrays are identical. Explain your answer. If you feel you need to make certain assumptions about the data, feel free to state them in your response.

Write your answer here, replacing this text.

Question 3.3 Set `biggest_range_dept` to the name of the department with the largest salary range, where range is defined as the **difference between the lowest and highest salaries in the department**.

Hint: First you'll need to define a new function `salary_range` which takes in an array of salaries and returns the salary range of the corresponding department. Then, set `department_ranges` to a table containing the names and salary ranges of each department.

In [ ]:

```
# Define salary_range in this cell
...
    ...
```

In [ ]:

```
department_ranges = ...
biggest_range_dept = ...
biggest_range_dept
```

4. Submission

Once you're finished running all cells, follow the instructions at the top of this notebook to save as a .pdf and then submit the file through Gradescope.