Assignment 2

2.8 (TABLE OF SQUARES AND CUBES) Write a script that calculates the squares and cubes of the numbers from 0 to 5. Print the resulting values in table format, as shown below. Use the tab escape sequence to achieve the three-column output.

```
number  square  cube
0       0       0
1       1       1
2       4       8
3       9       27
4       16      64
5       25      125
```

2.9 (INTEGER VALUE OF A CHARACTER) Here's a peek ahead. In this chapter, you learned about strings. Each of a string's characters has an integer representation. The set of characters a computer uses together with the characters' integer representations is called that computer's *character set*. You can indicate a character value in a program by enclosing that character in quotes, as in `'A'`. To determine a character's integer value, call the built-in function `ord`:

```
In [1]: ord('A')
Out[1]: 65
```

Display the integer equivalents of `B C D b c d 0 1 2 $ * +` and the space character.

2.10 (ARITHMETIC, SMALLEST AND LARGEST) Write a script that inputs three integers from the user. Display the sum, average, product, smallest and largest of the numbers. Note that each of these is a reduction in functional-style programming.

2.12 (7% INVESTMENT RETURN) Some investment advisors say that it's reasonable to expect a 7% return over the long term in the stock market. Assuming that you begin with $1000 and leave your money invested, calculate and display how much money you'll have after 10, 20 and 30 years. Use the following formula for determining these amounts:

$a = p(1 + r)^n$

where

p is the original amount invested (i.e., the principal of $1000),

r is the annual rate of return (7%),

n is the number of years (10, 20 or 30) and

a is the amount on deposit at the end of the

nth year.

Assignment 3

3.11 ***(Miles Per Gallon)*** Drivers are concerned with the mileage obtained by their automobiles. One driver has kept track of several tankfuls of gasoline by recording miles driven and gallons used for each tankful. Develop a sentinel-controlled-repetition script that prompts the user to input the miles driven and gallons used for each tankful. The script should calculate and display the miles per gallon obtained for each tankful. After processing all input information, the script should calculate and display the combined miles per gallon obtained for all tankfuls (that is, total miles driven divided by total gallons used).

Enter the gallons used (-1 to end): 12.8

Enter the miles driven: 287

The miles/gallon for this tank was 22.421875

Enter the gallons used (-1 to end): 10.3

Enter the miles driven: 200

The miles/gallon for this tank was 19.417475

Enter the gallons used (-1 to end): 5

Enter the miles driven: 120

The miles/gallon for this tank was 24.000000

Enter the gallons used (-1 to end): -1

The overall average miles/gallon was 21.601423

3.12 ***(Palindromes)*** A palindrome is a number, word or text phrase that reads the same backwards or forwards. For example, each of the following five-digit integers is a palindrome: 12321, 55555, 45554 and 11611. Write a script that reads in a five-digit integer and determines whether it's a palindrome. [*Hint*: Use the `//` and `%` operators to separate the number into its digits.]

3.13 ***(Factorials)*** Factorial calculations are common in probability. The factorial of a nonnegative integer n is written $n!$ (pronounced "n factorial") and is defined as follows:

$n! = n \cdot (n - 1) \cdot (n - 2) \cdot \ldots \cdot 1$

for values of n greater than or equal to 1, with 0! defined to be 1. So,

$5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$

which is 120. Factorials increase in size very rapidly. Write a script that inputs a nonnegative integer and computes and displays its factorial. Try your script on the integers 10, 20, 30 and even larger values. Did you find any integer input for which Python could not produce an integer factorial value?

4.13 (ARBITRARY ARGUMENT LIST) Calculate the product of a series of integers that are passed to the function `product`, which receives an arbitrary argument list. Test your function with several calls, each with a different number of arguments.

Assignment 4

1. **5.1 (WHAT'S WRONG WITH THIS CODE?)** What, if anything, is wrong with each of the following code segments?

```
day, high_temperature = ('Monday', 87, 65)
numbers = [1, 2, 3, 4, 5]
numbers[10]
name = 'amanda'
name[0] = 'A'
numbers = [1, 2, 3, 4, 5]
numbers[3.4]
student_tuple = ('Amanda', 'Blue', [98, 75, 87])
student_tuple[0] = 'Ariana'
('Monday', 87, 65) + 'Tuesday'
'A' += ('B', 'C')
x = 7
del x
print(x)
numbers = [1, 2, 3, 4, 5]
numbers.index(10)
numbers = [1, 2, 3, 4, 5]
numbers.extend(6, 7, 8)
numbers = [1, 2, 3, 4, 5]
numbers.remove(10)
values = []
```

```
values.pop()
```

5.5 (IPYTHON SESSION: SLICING) Create a string called `alphabet` containing `'abcdefghijklmnopqrstuvwxyz'`, then perform the following separate slice operations to obtain:

1. The first half of the string using starting and ending indices.
2. The first half of the string using only the ending index.
3. The second half of the string using starting and ending indices.
4. The second half of the string using only the starting index.
5. Every second letter in the string starting with `'a'`.
6. The entire string in reverse.
7. Every third letter of the string in reverse starting with `'z'`.

5.7 (DUPLICATE ELIMINATION) Create a function that receives a list and returns a (possibly shorter) list containing only the unique values in sorted order. Test your function with a list of numbers and a list of strings.

5.18 (SUMMING THE TRIPLES OF THE EVEN INTEGERS FROM 2 THROUGH 10) Starting with a list containing 1 through 10, use `filter`, `map` and `sum` to calculate the total of the triples of the even integers from 2 through 10. Reimplement your code with list comprehensions rather than `filter` and `map`.

Assignment 5

6.9 (DICTIONARY MANIPULATIONS) Using the following dictionary, which maps country names to Internet top-level domains (TLDs):

```
tlds = {'Canada': 'ca', 'United States': 'us', 'Mexico': 'mx'}
```

Perform the following tasks and display the results:

1. Check whether the dictionary contains the key `'Canada'`.
2. Check whether the dictionary contains the key `'France'`.
3. Iterate through the key–value pairs and display them in a two-column format.
4. Add the key–value pair `'Sweden'` and `'sw'` (which is incorrect).
5. Update the value for the key `'Sweden'` to `'se'`.
6. Use dictionary comprehension to reverse the keys and values.
7. With the result of part (f), use a dictionary comprehension to convert the country names to all uppercase letters.

6.10 (SET MANIPULATIONS) Using the following sets:

`{'red', 'green', 'blue'}`

`{'cyan', 'green', 'blue', 'magenta', 'red'}`

display the results of:

1. comparing the sets using each of the comparison operators.
2. combining the sets using each of the mathematical set operators.

Assignment 6

7.9 (INDEXING AND SLICING `ARRAY`S) Create an `array` containing the values `1`–`15`, `reshape` it into a 3-by-5 `array`, then use indexing and slicing techniques to perform each of the following operations:

1. Select row `2`.
2. Select column `5`.
3. Select rows `0` and `1`.
4. Select columns `2`–`4`.
5. Select the element that is in row 1 and column 4.
6. Select all elements from rows `1` and `2` that are in columns `0`, `2` and `4`.

7.23 (PANDAS: `DATAFRAME`S) Perform the following tasks with pandas `DataFrame`s:

1. Create a `DataFrame` named `temperatures` from a dictionary of three temperature readings each for `'Maxine'`, `'James'` and `'Amanda'`.
2. Recreate the `DataFrame` `temperatures` in Part (a) with custom indices using the `index` keyword argument and a list containing `'Morning'`, `'Afternoon'` and `'Evening'`.
3. Select from `temperatures` the column of temperature readings for `'Maxine'`.
4. Select from `temperatures` the row of `'Morning'` temperature readings.
5. Select from `temperatures` the rows for `'Morning'` and `'Evening'` temperature readings.
6. Select from `temperatures` the columns of temperature readings for `'Amanda'` and `'Maxine'`.

7. Select from `temperatures` the elements for `'Amanda'` and `'Maxine'` in the `'Morning'` and `'Afternoon'`.
8. Use the `describe` method to produce `temperatures`' descriptive statistics.
9. Transpose `temperatures`.
10. Sort `temperatures` so that its column names are in alphabetical order.

Assignment 7

8.16 (REGULAR EXPRESSIONS: LOCATING URLS) Use a regular expression to search through a string and to locate all valid URLs. For this exercise, assume that a valid URL has the form `http://www.`*domain_name*`.`*extension*, where *extension* must be two or more characters.

8.18 (REGULAR EXPRESSION: PASSWORD FORMAT VALIDATOR) Search online for secure password recommendations, then research existing regular expressions that validate secure passwords. Two examples of password requirements are:

1. Passwords must contain at least five words, each separated by a hyphen, a space, a period, a comma or an underscore.
2. Passwords must have a minimum of 8 characters and contain at least one each from uppercase characters, lowercase characters, digits and punctuation characters (such as characters in `'!@#$%<^>&*?'`).

Write regular expressions for each of the two requirements above, then use them to test sample passwords.

Assignment 8

9.3 (CLASS AVERAGE: WRITING STUDENT RECORDS TO A CSV FILE) An instructor teaches a class in which each student takes three exams. The instructor would like to store this information in a file named `grades.csv` for later use. Write code that enables an instructor to enter each student's first name and last name as strings and the student's three exam grades as integers. Use the `csv` module to write each record into the `grades.csv` file. Each record should be a single line of text in the following CSV format:

```
firstname,lastname,exam1grade,exam2grade,exam3grade
```

9.16 (WORKING WITH THE `DIAMONDS.CSV` DATASET IN PANDAS) In this book's data-science chapters, you'll work extensively with datasets, many in CSV format. You'll frequently use pandas to load datasets and prepare their data for use in machine-learning studies. Datasets are available for almost anything you'd want to study. There are numerous dataset repositories from which you can download datasets in CSV and other formats. In this chapter, we mentioned:

https://vincentarelbundock.github.io/Rdatasets/datasets.html

And

https://github.com/awesomedata/awesome-public-datasets

The Kaggle competition site:16

https://www.kaggle.com/datasets?filetype=csv

has approximately 11,000 datasets with over 7500 in CSV format. The U.S. government's `data.gov` site:

https://catalog.data.gov/dataset?res_format=CSV&_res_format_limit=0

has over 300,000 datasets with approximately 19,000 in CSV format.

In this exercise, you'll use the diamonds dataset to perform tasks similar to those you saw in the Intro to Data Science section. This dataset is available as `diamonds.csv` from various sources, including the Kaggle and Rdatasets sites listed above. The dataset contains information on 53,940 diamonds, including each diamond's carats, cut, color, clarity, depth, table (flat top surface), price and *x*, *y* and *z* measurements. The Kaggle site's web page for this dataset describes each column's content.

Assignment 9

The purpose of this assignment is to increase your exposure to the Python visualization landscape.

On the following web page, you will find 19 links to articles and videos about various Python visualization topics.

https://pyviz.org/overviews/index.html

- Browse all 19 links.
- Select the link that you find most interesting.
- Write a one-page summary about the article/video.
    - Include the link to article/video in your summary
    - Include why this article/video was most interesting
    - Include what you learned from the article/video
    - Do not add fluff or filler just to make it a full page. It is more important to say something meaningful than to achieve the full page.
- Re-create one of the visualizations in the article/video.
- Take a screenshot of the visualization.

Submit Summary, Visualization, and Code to Canvas.

Assignment 10

12.1 (WEB SCRAPING WITH THE REQUESTS AND BEAUTIFUL SOUP LIBRARIES)

Web pages are excellent sources of text to use in NLP tasks. In the following IPython session, you'll use the `requests` **library** to download the `www.python.org` home page's content. This is called **web scraping**. You'll then use the **Beautiful Soup library**[37] to extract only the text from the page. Eliminate the stop words in the resulting text, then use the `wordcloud` module to create a word cloud based on the text.

```
In [1]: import requests

In [2]: response = requests.get('https://www.python.org')

In [3]: response.content # gives back the page's HTML

In [4]: from bs4 import BeautifulSoup

In [5]: soup = BeautifulSoup(response.content, 'html5lib')

In [6]: text = soup.get_text(strip=True) # text without tags
```

In the preceding code, snippets `[1]`–`[3]` get a web page. The `get` **function** receives a URL as an argument and returns the corresponding web page as a `Response` object. The `Response`'s `content` **property** contains the web page's content. Snippets `[4]`–`[6]` get only the web page's text. Snippet `[5]` creates a `BeautifulSoup` object to process the text in `response.content`. `BeautifulSoup` method `get_text` with the keyword argument `strip=True` returns just the text of the web page without its structural information that your web browser uses to display the web page.

12.2 (TOKENIZING TEXT AND NOUN PHRASES) Using the text from Exercise 12.1, create a `TextBlob`, then tokenize it into `Sentence`s and `Word`s, and extract its noun phrases.

Assignment 11

The purpose of this assignment is to get experience data mining Twitter, applying preprocessing workflow and generate a word cloud with the resulting data.

Requirements:

- Create a Twitter developer account
- Get elevated access to API V2
- Using tweepy, retrieve a list of tweets. You can use any query you want.
- Use tweet-preprocessor to extract only the text from the tweet.
- Use NLTK and TextBlob to remove stop words and get word counts.
- Create a bar chart of the top 20 words. Can be less if your preprocessing leaves you with fewer than 20 words.
- Create a word cloud from your cleaned data.

Submit screenshot of developer account, code and visualizations to Canvas.

Files must be named using the following format.

LastName_FirstName_Week12_DeveloperAccount (Word or PDF)

LastName_FirstName_Week12_WordCloud (Word, PDF or Image)

LastName_FirstName_Week12_BarChart (Word, PDF or Image)

LastName_FirstName_Week12_Code (Script or Notebook)

Submit individual files to Canvas. Do not zip files.

Alternate Assignment

Report on Data Mining. Report can be about any Data Mining topic (Techniques, Tools, Algorithms, etc...). Rubric is below.

Submit paper to Canvas assignment named LastName_FirstName_Week12_DataMining (Word or PDF)

Alternative Assignment Rubric

Assignment 12

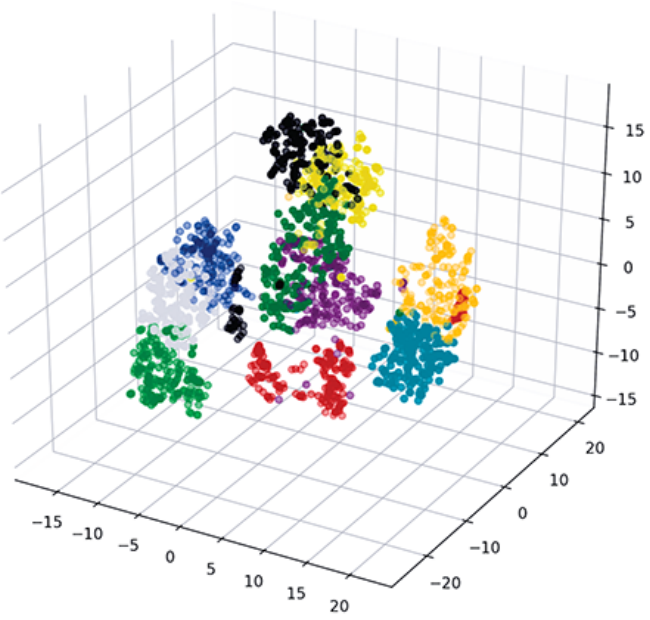
15.4 (HUMAN RECOGNITION OF HANDWRITTEN DIGITS) In this chapter, we analyzed the Digits dataset and used scikit-learn's `kNeighborsClassifier` to recognize the digits with high accuracy. Can humans recognize digit images as well as the `kNeighborsClassifier` did? Create a script that randomly selects and displays individual images and asks the user to enter a digit from 0 through 9 specifying the digit the image represents. Keep track of the user's accuracy. How does the user compare to the k-nearest neighbors machine-learning algorithm?

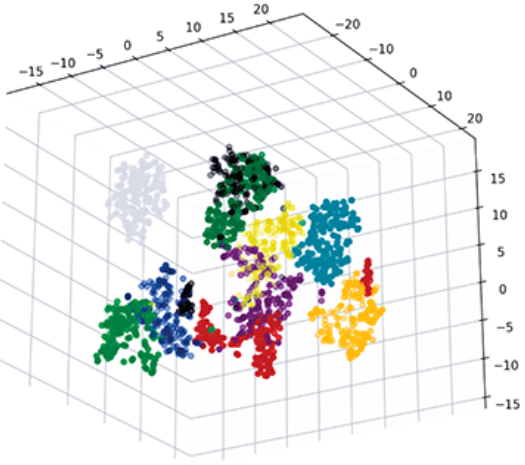
15.5 (USING `TSNE` TO VISUALIZE THE DIGITS DATASET IN 3D) In **Section 15.6**, you visualized the Digits dataset's clusters in two dimensions. In this exercise, you'll create a 3D scatter plot using `TSNE` and Matplotlib's `Axes3D`, which provides *x*-, *y*- and *z*-axes for plotting in three dimensions. To do so, load the Digits dataset, create a `TSNE` estimator that reduces data to three dimensions and call the estimator's `fit_transform` method to reduce the dataset's dimensions. Store the result in `reduced_data`. Next, execute the following code:

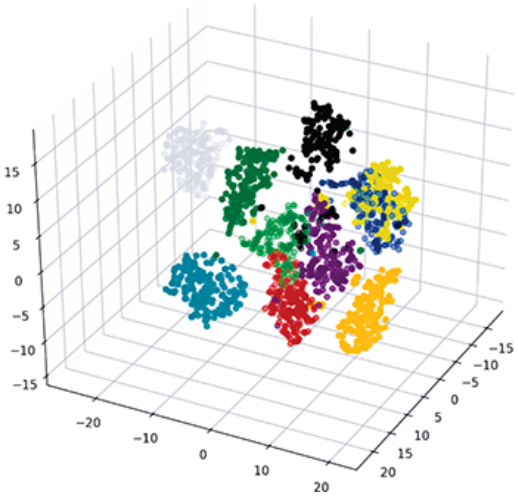
```
from mpl_toolkits.mplot3d import Axes3D

figure = plt.figure(figsize=(9, 9))

axes = figure.add_subplot(111, projection='3d')

dots = axes.scatter(xs=reduced_data[:, 0],
    ys=reduced_data[:, 1], zs=reduced_data[:, 2],
c=digits.target,
```

```
    cmap=plt.cm.get_cmap('nipy_spectral_r', 10))
```

The preceding code imports `Axes3D`, creates a `Figure` and calls its `add_subplot` method to get an `Axes3D` object for creating a three-dimensional graph. In the call to the `Axes3D` `scatter` method, the keyword arguments `xs`, `ys` and `zs` specify one-dimensional arrays of values to plot along the *x*-, *y*- and *z*-axes. Once the graph is displayed, be sure to drag the mouse on the image to rotate it left, right, up and down so you can see the clusters from various angles. The following images show the initial 3D graph and two rotated views:







15.6 (SIMPLE LINEAR REGRESSION WITH AVERAGE YEARLY NYC TEMPERATURES TIME SERIES) Go to NOAA's Climate at a Glance page (https://www.ncdc.noaa.gov/cag) and download the available time series data for the New York City average *annual* temperatures from 1895 through present (1895–2017 at the time of this writing). For your convenience, we provided the data in the file `ave_yearly_temp_nyc_1895-2017.csv`. Reimplement the simple linear regression case study of **Section 15.4** using the average yearly temperature data. How does the temperature trend compare to the average January high temperatures?

Use Machine Learning Python modules and algorithms to solve problems using data sets.

Start with textbook exercise 15.16 – Linear Regression with the California Housing Dataset

Dataset is available in sklearn using datasets.fetch_california_housing()

In addition to exercise requirements, you are required to complete the following items.

- Perform data exploration including 1 visualization
- Perform data cleanup
- Perform hyperparameter tuning
- Choose a second regression estimator and compare performance to Linear Regression
- Write a 500-word paper talking about the following
    - What produced better results, simple linear or multi linear regression?
    - What did you learn?
    - What challenges did you face?
    - Discuss performance difference of the two regression estimators

Submit Paper, Visualization and Code to Canvas.

Files must be named using the following format.

LastName_FirstName_FinalProject_Paper (Word or PDF)

LastName_FirstName_FinalProject_Visualization (Word, PDF or Image)

LastName_FirstName_FinalProject_Code (Script or Notebook)

Submit individual files to Canvas. Do not zip files.