Module 2 exercises:
Introduction to Python for Data Science

Exercise 1 – Virtual Environments

1. Connect to the environment called pds using Anaconda Navigator or Anaconda Prompt.
2. Check the virtual environment has the following packages installed:
    a. Numpy
    b. Pandas
    c. Seaborn
    d. Matplotlib
    e. scikit-learn

Exercise 2 – Jupyter Notebooks

1. Create a new notebook.

Experiment with the keyboard shortcuts in Command mode. You might want to try:

- Adding cells above and below.
- Deleting cells
- Changing from markdown to code.
- Verifying, i.e., entering some text or code and running checks.

Exercise 3 – Python Review

3.1 – Display a message using variables

1. Create two variables to hold a person's name and age.
2. Display the person's name and age with a space in between.

```
username='Bob'
age=32
print(username,'is',age,'years old')
```

```
Bob is 32 years old
```

3. Alter the code so that . is between the person's name and age.

```
username='Bob'
age=32
print(username,'is',age,'years old', sep='.')
```

```
Bob.is.32.years old
```

4. Alter the code so that the person's name is displayed in uppercase.

```
username='Bob'
age=32
print(username.upper(),'is',age,'years old', sep='.')
```

```
BOB.is.32.years old
```

3.2 – Casting variables

1. Capture user input to get the length of the first side of a rectangle.

   Use a suitable variable name such as length.

   You must cast (convert) the text you input to an integer type (int).

```
length = int(input("What is the length of the first side? "))
```

```
What is the length of the first side?  4
```

2. Input the length of the second side of the rectangle.

   Use a suitable variable name such as width.

   Again, cast the input text to an integer type (int).

```
width = int(input("What is the length of the second side? "))
```

```
What is the length of the second side?  4
```

3. Calculate and display the perimeter of the rectangle.

```
print(f'The rectangle with length {length} and width {width} has perimeter {2 * (length + width)}')
```

```
The rectangle with length 4 and width 4 has perimeter 16
```

3.3 – Data structures

Using the below data structures, complete the following tasks:

```
ages = [12,18,33,84,45,67,12,82,95,16,10,23,43,29,40,34,30,16,44,69,70
,74,38,65,36,83,50,11,79,64,78,37,3,8,68,22,4,60,33,82,45,23,5,18,28,9
9,17,81,14,88,50,19,59,7,44,93,35,72,25,63,11,69,11,76,10,60,30,14,21,
82,47,6,21,88,46,78,92,48,36,28,51]
byron = 'Fare thee well and if forever still forever fare thee well'
desserts = 'sticky toffee pudding', 'affogato', 'syrup sponge'
books = {
  "title": "Hamlet",
  "author": "William Shakespeare",
  "language": "English"
}
```

- Display the length of the ages list.

```
age_length = len(ages)
print(age_length)
```

```
81
```

- Display the ages from the third position to the last.

```
print(ages[3:])
```

```
[84, 45, 67, 12, 82, 95, 16, 10, 23, 43, 29, 40, 34, 30, 16, 44, 69, 70, 74, 38, 65, 36, 83, 50, 11, 79, 64, 78, 3
7, 3, 8, 68, 22, 4, 60, 33, 82, 45, 23, 5, 18, 28, 99, 17, 81, 14, 88, 50, 19, 59, 7, 44, 93, 35, 72, 25, 63, 11,
69, 11, 76, 10, 60, 30, 14, 21, 82, 47, 6, 21, 88, 46, 78, 92, 48, 36, 28, 51]
```

- Find the position of the first occurrence of the word 'forever' in Byron.

```
byron.find('forever')
```

```
22
```

- Print the second of the desserts.

```
desserts[1]
```

```
'affogato'
```

- Check if pancakes are included in desserts.

```
'pancakes' in desserts
```

```
False
```

- Print the title value from books.

```
books['title']
```

```
'Hamlet'
```

3.4 – Functions

1. Write a function that takes a string as an argument and returns the number of capital letters in the string. Hint: 'foo'.upper() returns 'FOO'.

```
def make_upper(text: str) -> None:
    return text.upper()
```

2. Write a function that takes two sequences seq_a and seq_b as arguments and returns True if every element in seq_a is also an element of seq_b, else False.

   By "sequence" we mean a list, a tuple or a string.
   Do the exercise without using sets and set methods.

```
def seq_check(seq_a, seq_b):
    return all([seq_a_element in seq_b for seq_a_element in seq_a])
```

3.5 – (Extension) Iteration

Write a piece of code which calculates change to be given from an input amount in £. To start you off, we've given you a list of denominations of UK currency.

```
denoms = [50, 20, 10, 5, 2, 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01]
```

```
denoms = [50, 20, 10, 5, 2, 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01]
amount = float(input("What do you need change from? £"))

numbers = []
for denom in denoms:
    numbers.append(int(amount // denom))
    if amount % denom == 0:
        break
    else:
        amount = round(amount % denom, 2) # There are better types for this

for num, denom in zip(numbers, denoms):
    print(f"{num} x £{denom}")
```

```
What do you need change from? £ 5
0 x £50
0 x £20
0 x £10
1 x £5
```

Exercise 4 – Numpy

```
# import necessary libraries
import numpy as np

# create data to work with
x = np.arange(10)
x
```

1. Display the first 5 elements.

```
x[:5]
```

```
array([0, 1, 2, 3, 4])
```

2. Display every other element.

```
x[::2]
```

```
array([0, 2, 4, 6, 8])
```

3. Display the elements from index 5 in reverse order.

```
# two dimensional array
y = np.array([[12,  5,  2,  4],
              [ 7,  6,  8,  8],
              [ 1,  6,  7,  7]])
```

```
x[5::-1]
```

```
array([5, 4, 3, 2, 1, 0])
```

4. Display the first 2 rows and the first 3 columns of y.

```
y[:2, :3]
```

```
array([[12,  5,  2],
       [ 7,  6,  8]])
```

5. Read the contents of file cdc.csv, containing heights, weights and ages, into array data. To do this, you can use the below code:

```
data = np.genfromtxt('data/cdc.csv', delimiter=',',
skip_header=1)
```

6. Separate the heights (column 5) and the weights (column 6)

```
heights = data[:,5]
```

```
weights = data[:,6]
```

7. Calculate the median for the heights and the weights and assign the values to variables.

```
median_heights = np.median(heights)
median_heights
```

```
67.0
```

```
median_weights = np.median(weights)
median_weights
```

```
165.0
```

Exercise 5 – Pandas

1. Read the file mortgage_applicants.csv, which sits in the data folder, into a variable called mortgage. **Hint: Your path will need to reflect the location of the file.**

```
import pandas as pd

mortgage = pd.read_csv('data/mortgage_applicants.csv')
mortgage
```

	Unnamed: 0	ID	Income	Term	Balance	Debt	Score	Default
0	0	567	17626	10 Years	1381	293	228.0	False
1	1	523	18959	20 Years	883	1012	187.0	False
2	2	544	20560	10 Years	684	898	86.0	False
3	3	370	21894	10 Years	748	85	NaN	False
4	4	756	24430	10 Years	1224	59	504.0	False
...	...	...	...	...	...	...	...	...

2. Select the Score column of the mortgage DataFrame

```
mortgage['Score']
```

```
0      228.0
1      187.0
2       86.0
3        NaN
4      504.0
       ...
851     55.0
852    780.0
853    366.0
854    422.0
855    179.0
Name: Score, Length: 856, dtype: float64
```

3. Select the Balance and Income columns.

```
mortgage[['Balance', 'Income']]
```

	Balance	Income
0	1381	17626
1	883	18959
2	684	20560
3	748	21894
4	1224	24430
...	...	...

4. Select the first row in the mortgage DataFrame.

```
mortgage.iloc[0, :]
```

```
Unnamed: 0          0
ID                567
Income          17626
Term         10 Years
Balance          1381
Debt              293
Score           228.0
Default         False
Name: 0, dtype: object
```

5. Display all mortgage applicants who have a Balance greater than £1000.

```
mortgage[mortgage['Balance'] > 1000]
```

	Unnamed: 0	ID	Income	Term	Balance	Debt	Score	Default
0	0	567	17626	10 Years	1381	293	228.0	False
4	4	756	24430	10 Years	1224	59	504.0	False
5	5	929	22995	20 Years	1678	1329	384.0	False
6	6	373	21124	10 Years	1135	115	560.0	False
7	7	818	24644	10 Years	1634	105	309.0	False
...	...	...	...	...	...	...	...	...

6. Display all mortgage applicants who have a Balance greater than £1000 and a Debt below £50.

```
mortgage[(mortgage['Balance'] > 1000) & (mortgage['Debt'] < 50)]
```

	Unnamed: 0	ID	Income	Term	Balance	Debt	Score	Default
12	12	327	18630	20 Years	1316	37	430.0	False
54	54	608	25267	10 Years	1308	2	470.0	False
56	56	67	20528	10 Years	1253	40	511.0	False
59	59	316	16142	10 Years	1251	12	281.0	False
66	66	472	19400	10 Years	1292	24	235.0	False
...	...	...	...	...	...	...	...	...

Exercise 6 – Visualistion

1. Load in the dataset train_viz.csv

```
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('data/train_viz.csv')
```

2. Create a scatter plot of duration vs. price. Segment by destination.

```
sns.scatterplot(data=df,
                x='duration',
                y='price',
                hue='destination');
```