Exercises_wk3 (including solutions)

Q1: Go to the following website (http://www.gutenberg.org/files/1342/1342-0.txt), and save this page as a text-file ('pride_and_prejudice.txt') in the same folder that you are in when you start Jupyter Notebook. Then read this whole text-file into one string using the read() method on the file handle and print the characters [2062:2612] of this string. (Hint: the first line of code should probably be something like: fhand = open('C:\...\pride_and_prejudice.txt', 'r', encoding="utf8"), depending on where you have stored the text-file.)

NOTE: If necessary, you can use the following code to identify the path of the working directory where Python is currently running:

```
import os
os.getcwd()
```

In [1]:
```
fhand = open('C:\\Users\Martijn\Desktop\Supporting website\pride_and_prejudice.txt', 'r', enco
ding="utf8")
text = fhand.read()
print(text[2062:2612])

fhand.close()
```

```
Chapter 1

      It is a truth universally acknowledged, that a single man in
      possession of a good fortune, must be in want of a wife.

      However little known the feelings or views of such a man may be
      on his first entering a neighbourhood, this truth is so well
      fixed in the minds of the surrounding families, that he is
      considered the rightful property of some one or other of their
      daughters.

      “My dear Mr. Bennet,” said his lady to him one day, “have you
      heard that Netherfield Park is let at last?”
```

NOTE: A list is a sequence of values, which can be any data type (even other (nested) lists). Lists are mutable.

Q2: Define a list ('list1') that contains the following values: 'Amazon', [2019, 2020], 20.5, 798000, and print it. Then print the length of 'list1'.

In [2]:
```
list1 = ['Amazon', [2019, 2020], 20.5, 798000]
print(list1)
print(len(list1))
```

```
['Amazon', [2019, 2020], 20.5, 798000]
4
```

Q3: Use indexing and slicing to respectively extract: (1) the element at index position 0 of the list ('Amazon'), (2) the element at index position 1 of the list ('[2019, 2020]'), (3) the element at index position 1 of the nested list ('2020'), and (4) the slice from the element at index position 2 to the end of the list ('[20.5, 798000]'), and print these elements and slices. (Hint: Be aware that Python uses zero-based indexing.)

In [3]:
```
print(list1[0])
print(list1[1])
print(list1[1][1])
print(list1[2:])
```

```
Amazon
[2019, 2020]
2020
[20.5, 798000]
```

NOTE: When working with loops in combination with lists, programs can often be shortened by using something that is called list comprehension. For example, in order to generate (and print) a random list of 10 numbers between 5 and 25 (using the built-in random module), instead of using the following for loop:

```
import random

list_of_numbers = [] # Start with an empty list
for i in range(10):
    num = random.randint(5, 25) # Here the random numbers are generated
    list_of_numbers.append(num) # Here the generated numbers are added to the list
print(list_of_numbers)
```

, also the following program that uses list comprehension can be used:

```
import random

list_of_numbers = [random.randint(5, 25) for i in range(10)]
print(list_of_numbers)
```

There are many do's and don'ts for list comprehension. Just Google a little to find more information about it. (Note: Given that they generate a random list of numbers, these two programs will of course (almost certainly) not generate the exact same list. However, you can use the random.seed() function to save the state of the random function, so that it generates the same random numbers on multiple executions of the code.)

Q4: Copy-paste one of the two programs from the note above into a cell to generate and print a list of 10 random numbers between 5 and 25.

In [4]:
```
import random

list_of_numbers = []
# random.seed(123)
for i in range(10):
    num = random.randint(5, 25)
    list_of_numbers.append(num)
print(list_of_numbers)
```

```
[9, 21, 17, 9, 24, 13, 18, 20, 19, 25]
```

Q5: Calculate the sum and the number of elements of this list, and assign each of them to a variable. (Hint: Use the sum() and len() functions.) Next, use these two variables to calculate the mean of the list, and print the sum, the length and the mean of the list on the same line.

In [5]:
```
lst_sum = sum(list_of_numbers)
lst_len = len(list_of_numbers)
lst_mean = lst_sum / lst_len
print(lst_sum, lst_len, lst_mean)
```

```
175 10 17.5
```

Q6: Make a copy of the list ('list_min_max') and then sort this copied list from minimum to maximum, and print it. Next, do the same for maximum to minimum (i.e., in reversed order).

In [6]:
```
list_min_max = list_of_numbers.copy()
list_min_max.sort()
print (list_min_max)

list_max_min = list_of_numbers.copy()
list_max_min.sort(reverse = True)
print (list_max_min)
```

```
[9, 9, 13, 17, 18, 19, 20, 21, 24, 25]
[25, 24, 21, 20, 19, 18, 17, 13, 9, 9]
```

Q7: Define a list ('largest_firms') that contains the following firms (which are the ten largest companies by revenue in 2020, mostly for FY 2019, according to the Fortune 1000; see https://en.wikipedia.org/wiki/List_of_largest_companies_in_the_United_States_by_revenue): 'Walmart', 'Amazon', 'ExxonMobil', 'Apple', 'CVS Health', 'Berkshire Hathaway', 'UnitedHealth Group', 'McKesson', 'AT&T', 'AmerisourceBergen', and print it.

In [7]:
```
largest_firms = ['Walmart', 'Amazon', 'ExxonMobil', 'Apple', 'CVS Health', 'Berkshire Hathaway
', 'UnitedHealth Group', \
                 'McKesson', 'AT&T', 'AmerisourceBergen']
print(largest_firms)
```

```
['Walmart', 'Amazon', 'ExxonMobil', 'Apple', 'CVS Health', 'Berkshire Hathaway', 'UnitedHealt
h Group', 'McKesson', 'AT&T', 'AmerisourceBergen']
```

Q8: Extend the list with the next five firms ('Alphabet', 'Ford', 'Cigna', 'Costco', 'Chevron'), so that it now includes the fifteen largest companies by revenue in 2020, mostly for FY 2019, according to the Fortune 1000, and print it again. Then print the length of 'largest_firms'.

In [8]:
```
largest_firms.extend(['Alphabet', 'Ford', 'Cigna', 'Costco', 'Chevron'])
print(largest_firms)
print(len(largest_firms))
```

```
['Walmart', 'Amazon', 'ExxonMobil', 'Apple', 'CVS Health', 'Berkshire Hathaway', 'UnitedHealt
h Group', 'McKesson', 'AT&T', 'AmerisourceBergen', 'Alphabet', 'Ford', 'Cigna', 'Costco', 'Ch
evron']
15
```

Q9: Create a new list ('largest_firms_low') with the names of all firms from 'largest_firms' written in lowercase letters. Then sort this new list in alphabetical order, and print it. (Hint: Use the lower() method and the sort() method.)

In [9]:
```
largest_firms_low = []
for item in largest_firms:
    largest_firms_low.append(item.lower())
largest_firms_low.sort()
print(largest_firms_low)
```

```
['alphabet', 'amazon', 'amerisourcebergen', 'apple', 'at&t', 'berkshire hathaway', 'chevron',
'cigna', 'costco', 'cvs health', 'exxonmobil', 'ford', 'mckesson', 'unitedhealth group', 'wal
mart']
```

Q10: Use indexing to extract the firm at index position 3 ('apple'), and print it. Next, use slicing to extract the slice from the firm at index position 6 up to (but not including) the firm at index position 9 ('chevron', 'cigna', 'costna'), and print this slice.

In [10]:
```
print(largest_firms_low[3])
print(largest_firms_low[6:9])
```

```
apple
['chevron', 'cigna', 'costco']
```