Weeks 1&2 Assignment

November 29, 2020

DSC540 Michael Hotaling Weeks 1 & 2 2020-11-28

[1]:
```
import pandas as pd
import numpy as np
import random as rd
import matplotlib.pyplot as plt
from jupyterthemes import jtplot
from IPython.core.display import display, HTML
# jtplot.style(theme='onedork')
```

1) Create a Jupyter notebook where you:

a) Create a list

[2]:
```
my_list = [6, 2, 1, 4, 9, 6, 3, 8, 13, 10]
my_list
```

[2]:
```
[6, 2, 1, 4, 9, 6, 3, 8, 13, 10]
```

b) Iterate over the list to sort your results

[3]:
```
for i in np.arange(0, len(my_list)):
    key = my_list[i]
    j = i-1
    while j >=0 and key < my_list[j] :
            my_list[j+1] = my_list[j]
            j += -1
    my_list[j+1] = key

print(my_list)
```

```
[1, 2, 3, 4, 6, 6, 8, 9, 10, 13]
```

c) Generate random numbers

[4]:
```
rand_nums = []

for i in range(0,10):
    n = rd.randint(1,20)
    rand_nums.append(n)
```

```
print(rand_nums)
```

```
[12, 9, 4, 20, 2, 16, 3, 5, 8, 14]
```

d) Add to the list, and then print your results.

[5]:
```
my_list = my_list + rand_nums
print(my_list)
```

```
[1, 2, 3, 4, 6, 6, 8, 9, 10, 13, 12, 9, 4, 20, 2, 16, 3, 5, 8, 14]
```

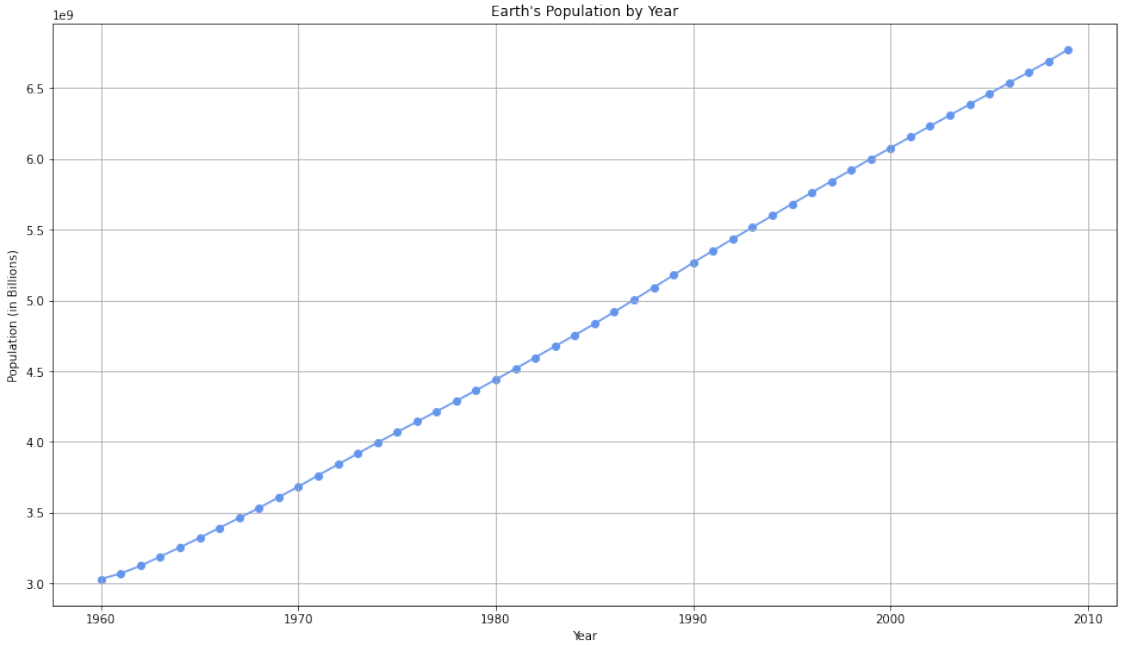
2) Create a line chart with Matplotlib and the following data file.

[6]:
```
df = pd.read_excel("world-population.xlsm")
df.head()
```

[6]:

	Year	Population
0	1960	3028654024
1	1961	3068356747
2	1962	3121963107
3	1963	3187471383
4	1964	3253112403

[7]:
```
plt.figure(figsize=(16, 9))
plt.plot(df['Year'], df['Population'], color = "cornflowerblue")
plt.scatter(df['Year'], df['Population'], color = "cornflowerblue");
plt.title("Earth's Population by Year")
plt.xlabel("Year")
plt.grid()
plt.ylabel("Population (in Billions)");
```



0.0.1 Activity 1: Handling Lists

In this activity, we will generate a **list** of random numbers and then generate another **list** from the first one, which only contain numbers that are divisible by three. Repeat the experiment three times Then, we will calculate the average difference of length between the two lists. These are the steps for completing this activity:

1) Create a `list` of 100 random numbers.

[8]:
```
def generate_random_numbers(l = 100):
    """Return a list of random numbers. Default is 100."""
    random_numbers = []
    for i in range(0,l):
        random_numbers.append(rd.randint(1,100))
    return random_numbers

rand_nums = generate_random_numbers()
print(rand_nums)
```

```
[93, 72, 68, 30, 84, 62, 50, 30, 74, 47, 95, 98, 7, 75, 8, 77, 7, 85, 15, 9, 73,
19, 3, 65, 5, 31, 24, 68, 30, 17, 22, 45, 89, 22, 55, 97, 49, 40, 84, 57, 54,
19, 83, 6, 77, 75, 91, 79, 27, 83, 47, 27, 59, 22, 84, 73, 58, 29, 40, 92, 27,
59, 52, 62, 100, 33, 65, 69, 21, 37, 63, 58, 49, 96, 79, 81, 37, 31, 67, 34, 6,
79, 26, 76, 9, 11, 18, 1, 55, 80, 82, 91, 23, 50, 1, 85, 97, 90, 75, 52]
```

2) Create a new `list` from this random `list`, with numbers that are divisible by 3.

[9]:
```
def divisible_by_three(arr):
    """Intakes a list and returns a new list containing only values in the␣
 ↪original list that are divisible by three"""
    new_list = []
    for i in arr:
        if i % 3 == 0:
            new_list.append(i)
    return new_list

new_list = divisible_by_three(rand_nums)
print(new_list)
```

```
[93, 72, 30, 84, 30, 75, 15, 9, 3, 24, 30, 45, 84, 57, 54, 6, 75, 27, 27, 84,
27, 33, 69, 21, 63, 96, 81, 6, 9, 18, 90, 75]
```

3) Calculate the length of these two lists and store the difference in a new variable

[10]:
```
len_rand_list = len(rand_nums)
len_new_list = len(new_list)
list_diff = len_rand_list - len_new_list
```

```
print("Random Number List Length: {}".format(len_rand_list))
print("New List Length: {}".format(len_new_list))
print("Diff is: {}".format(list_diff))
```

```
Random Number List Length: 100
New List Length: 32
Diff is: 68
```

4) Using a loop, perform steps 2 and 3 and find the difference variable three times

[11]:
```
# In order to get different results, I'll need to regenerate the random numbers␣
 ↪list,
# else I'll just get the same return

runs = 3

diff_list_len = []

for i in range(0,runs):
    first_list = generate_random_numbers()
    second_list = divisible_by_three(first_list)
    diff_list_len.append(len(first_list) - len(second_list))

print("Diff List: {}".format(diff_list_len))
```

```
Diff List: [74, 69, 64]
```

5) Find the arithmetic mean of these three difference values

[12]:
```
def mean(arr, digits = 2):
    """Calculates the arithmetic mean of an array"""
    summer = 0
    for i in arr:
        summer += i
    return round(summer/len(arr), digits)

mean(diff_list_len)
```

[12]: 69.0

0.0.2 Activity 2: Analyze a Multiline String and Generate the Unique Word Count

This section will ensure that you have understood the various basic data structrues and their manipulation. We will do that by going though an activity that has been designed specifically for this purpose: In this activity, we will do the following: - Get multiline text and save it in a Python variable - Get rid of all new lines in it using string methods - Get all the unique words and their occurences from the string - Repeat the step to find all unique words and occurrences, without considering case sensitivity

> **Note** For the sake of simpliity for this activity, the original text (which can be found at https://www.gutenberg.org/files/1342/1342-h/1342-h.htm) has been pre-processed a bit

These are the steps to guide you through solving this activity: **1) Create a `multiline_text` variable by copying the text from the first chapter of *Pride and Prejudice*.**

> **Note** The first chapter of *Pride and Prejudice* by Jane Austen has been made available on the GitHub repository at http://github.com/TrainingByPackt/Data-Wrangling-with-Python/blob/master/Chapter01/Activity02/

[13]:
```
import string
```

[14]:
```
multiline_text = """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?"

Mr. Bennet replied that he had not.

"But it is," returned she; "for Mrs. Long has just been here, and she told me␣
 ↪all about it."

Mr. Bennet made no answer.

"Do you not want to know who has taken it?" cried his wife impatiently.

"You want to tell me, and I have no objection to hearing it."

This was invitation enough.

"Why, my dear, you must know, Mrs. Long says that Netherfield is taken by a␣
 ↪young man of large fortune from the north of England; that he came down on␣
 ↪Monday in a chaise and four to see the place, and was so much delighted with␣
 ↪it, that he agreed with Mr. Morris immediately; that he is to take␣
 ↪possession before Michaelmas, and some of his servants are to be in the␣
 ↪house by the end of next week."

"What is his name?"

"Bingley."
```

```
"Is he married or single?"

"Oh! Single, my dear, to be sure! A single man of large fortune; four or five␣
 ↪thousand a year. What a fine thing for our girls!"

"How so? How can it affect them?"

"My dear Mr. Bennet," replied his wife, "how can you be so tiresome! You must␣
 ↪know that I am thinking of his marrying one of them."

"Is that his design in settling here?"

"Design! Nonsense, how can you talk so! But it is very likely that he may fall␣
 ↪in love with one of them, and therefore you must visit him as soon as he␣
 ↪comes."

"I see no occasion for that. You and the girls may go, or you may send them by␣
 ↪themselves, which perhaps will be still better, for as you are as handsome␣
 ↪as any of them, Mr. Bingley may like you the best of the party."

"My dear, you flatter me. I certainly have had my share of beauty, but I do not␣
 ↪pretend to be anything extraordinary now. When a woman has five grown-up␣
 ↪daughters, she ought to give over thinking of her own beauty."

"In such cases, a woman has not often much beauty to think of."

"But, my dear, you must indeed go and see Mr. Bingley when he comes into the␣
 ↪neighbourhood."

"It is more than I engage for, I assure you."

"But consider your daughters. Only think what an establishment it would be for␣
 ↪one of them. Sir William and Lady Lucas are determined to go, merely on that␣
 ↪account, for in general, you know, they visit no newcomers. Indeed you must␣
 ↪go, for it will be impossible for us to visit him if you do not."

"You are over-scrupulous, surely. I dare say Mr. Bingley will be very glad to␣
 ↪see you; and I will send a few lines by you to assure him of my hearty␣
 ↪consent to his marrying whichever he chooses of the girls; though I must␣
 ↪throw in a good word for my little Lizzy."

"I desire you will do no such thing. Lizzy is not a bit better than the others;␣
 ↪and I am sure she is not half so handsome as Jane, nor half so good-humoured␣
 ↪as Lydia. But you are always giving her the preference."
```

```
"They have none of them much to recommend them," replied he; "they are all␣
 ↪silly and ignorant like other girls; but Lizzy has something more of␣
 ↪quickness than her sisters."

"Mr. Bennet, how can you abuse your own children in such a way? You take␣
 ↪delight in vexing me. You have no compassion for my poor nerves."

"You mistake me, my dear. I have a high respect for your nerves. They are my␣
 ↪old friends. I have heard you mention them with consideration these last␣
 ↪twenty years at least."

"Ah, you do not know what I suffer."

"But I hope you will get over it, and live to see many young men of four␣
 ↪thousand a year come into the neighbourhood."
""
"It will be no use to us, if twenty such should come, since you will not visit␣
 ↪them."

"Depend upon it, my dear, that when there are twenty, I will visit them all."

Mr. Bennet was so odd a mixture of quick parts, sarcastic humour, reserve, and␣
 ↪caprice, that the experience of three-and-twenty years had been insufficient␣
 ↪to make his wife understand his character. Her mind was less difficult to␣
 ↪develop. She was a woman of mean understanding, little information, and␣
 ↪uncertain temper. When she was discontented, she fancied herself nervous.␣
 ↪The business of her life was to get her daughters married; its solace was␣
 ↪visiting and news. """
```

2) Find the type and length of the `multiline_text` string using the commands `type` and `len`

[15]:
```
type(multiline_text)
```

[15]:
```
str
```

[16]:
```
len(multiline_text)
```

[16]:
```
4478
```

3) Remove all new lines and symbols using the `replace` function.

[17]:
```
multiline_text = multiline_text.replace("\n", " ")
multiline_text = multiline_text.translate(multiline_text.maketrans("", "",␣
 ↪string.punctuation))#.lower()

multiline_text[:116]
```

[17]:
```
'It is a truth universally acknowledged that a single man in possession of a
good fortune must be in want of a wife  '
```

4) Find all the words in `multiline_text` using the `split` function.

[18]:
```
multiline_text = multiline_text.split(" ")
multiline_text = [i for i in multiline_text if i]
print(multiline_text[:26])
```

```
['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']
```

5) Create a list from this list that will contain only the unique words.

[19]:
```
new_list = []
for i in multiline_text:
    if i not in new_list:
        new_list.append(i)
```

[20]:
```
# We can compare the list to the set which removes all duplicate values to␣
↪verify our method worked
len(new_list) == len(set(multiline_text))
```

[20]:
```
True
```

6) Count the number of times the unique word has appeared in the list using the `key` and `value` in `dict`

[21]:
```
word_dictionary = dict()

for word in multiline_text:
    if word in word_dictionary:
        word_dictionary[word] += 1
    else:
        word_dictionary[word] = 1
```

7) Find the top 25 words from the unique words that you have found using the `slice` function

[22]:
```
# I have no idea how slice is suppose to work for this exercise. The solution␣
↪doesn't even use it

print("-" * 35)
print("|{:^33}|".format("Pride and Prejudice"))
print("-" * 35)
for index, key in enumerate(sorted(word_dictionary, key=word_dictionary.get,␣
↪reverse=True),start = 1):
    print("| {:>5} | {:15}|{:>7} |".format(index, key, word_dictionary[key]))
    if index == 25:
```

```
        break
print("-" * 35)
```

```
-----------------------------------
|       Pride and Prejudice       |
-----------------------------------
|     1 | of              |     29 |
|     2 | you             |     24 |
|     3 | to              |     22 |
|     4 | a               |     20 |
|     5 | the             |     17 |
|     6 | I               |     17 |
|     7 | and             |     16 |
|     8 | that            |     15 |
|     9 | is              |     12 |
|    10 | for             |     12 |
|    11 | in              |     11 |
|    12 | be              |     11 |
|    13 | his             |     11 |
|    14 | he              |     11 |
|    15 | it              |     11 |
|    16 | them            |     11 |
|    17 | Mr              |     10 |
|    18 | my              |     10 |
|    19 | not             |      9 |
|    20 | will            |      9 |
|    21 | so              |      8 |
|    22 | dear            |      8 |
|    23 | was             |      8 |
|    24 | are             |      8 |
|    25 | must            |      7 |
-----------------------------------
```

0.0.3 Activity 3: Permutation, Iterator, Lambda LIst

In this activity, we will be using **permutations** to generate all possible three-digit numbers that can be generated using 0, 1, and 2. Then, loop over this iterator, and also use `isinstance` and `assert` to make sure that the return types are tuples. Also, use a signle line of code involving `dropwhile` and `lambda` expressions to convert all the tuples to lists while dropping an leading zeros (for example, (0,1,2) becomes [1, 2]). Finally, write a function that takes a list like b efore and returns the actual number contained in it.

1) Look up the definitions of `permutation` and `dropwhile` from itertools

[23]:
```
from itertools import permutations, dropwhile
```

[24]:
```
?permutations
```

Init signature: permutations(iterable, r=None)

Docstring:
Return successive r-length permutations of elements in the iterable.

[25]:
```
?dropwhile
```

Init signature: dropwhile(predicate, iterable, /)

Docstring:
Drop items from the iterable while predicate(item) is true.

2) Write and expression to generate all the possible three-digit numbers using 0, 1, and 2.

[26]:
```
def number_combiner(arr):
    nums = permutations(arr, len(arr))
    num_list = []
    for i in nums:
        num_list.append(i)
    return num_list
```

[27]:
```
my_nums = [0, 1, 2]
number_combiner(my_nums)
```

[27]:
```
[(0, 1, 2), (0, 2, 1), (1, 0, 2), (1, 2, 0), (2, 0, 1), (2, 1, 0)]
```

3) Loop over the iterator expression you generated before. Print each element that's returned by the iterator. Use `assert` and `isinstance` to make sure that the elements are of the tuple type

[28]:
```
for i in number_combiner(my_nums):
    assert type(i) == tuple
    print("| {:>7} | {:>10} | {:>5} |".format(str(i), str(type(i)),␣
 ↪str(isinstance(i, tuple))))
```

```
| (0, 1, 2) | <class 'tuple'> |  True |
| (0, 2, 1) | <class 'tuple'> |  True |
| (1, 0, 2) | <class 'tuple'> |  True |
| (1, 2, 0) | <class 'tuple'> |  True |
| (2, 0, 1) | <class 'tuple'> |  True |
| (2, 1, 0) | <class 'tuple'> |  True |
```

4) Write the loop again using `dropwhile` with a lambda expression to drop any leading zeros from the tuples. As an example, (0, 1, 2) will become (1, 2). Also, cast the output of dropwhile to a list.

[29]:
```
for i in number_combiner(my_nums):
    adj_list = list(dropwhile(lambda x: x == 0, i))
    print("| {:^10} | {:>10} | {:>5} |".format(str(adj_list),
                                               str(type(adj_list)),
                                               str(isinstance(adj_list, list))))
```

```
|   [1, 2]   | <class 'list'> |  True |
|   [2, 1]   | <class 'list'> |  True |
| [1, 0, 2]  | <class 'list'> |  True |
| [1, 2, 0]  | <class 'list'> |  True |
| [2, 0, 1]  | <class 'list'> |  True |
| [2, 1, 0]  | <class 'list'> |  True |
```

5) Check the actual tupe that `dropwhile` returns

[30]:
```
print(type(dropwhile(1,[1,2])))
```

```
<class 'itertools.dropwhile'>
```

6) Combine the preceding code into one block, and this time write a separate function where you will pass the list generated from `dropwhile`, and the function will return the whole number contained in the list. As an example, if you pass [1, 2] to the function, it will return 12. Make sure that the return type is indeed a number and not a string. Although, this task can be achieved using other tricks, we require that you treat the incoming list as a stack in the function and generate the number by reading the individual digits from the stack.

[31]:
```
for i in number_combiner(my_nums):
    adj_list = list(dropwhile(lambda x: x <= 0, i))
    new_nums = []
    temp_string = ""
    for j in adj_list:
        temp_string += str(j)
    temp_string = int(temp_string)
    new_nums.append(temp_string)
    print("| {:^10} | {:>10} | {:>5} |".format(str(temp_string),
                                               str(type(temp_string)),
                                               str(isinstance(temp_string,␣
 ↪int))))
```

```
|     12     | <class 'int'> |  True |
|     21     | <class 'int'> |  True |
|    102     | <class 'int'> |  True |
|    120     | <class 'int'> |  True |
|    201     | <class 'int'> |  True |
|    210     | <class 'int'> |  True |
```

0.0.4 Activity 4: Design Your Own CSV Parser

A CSV file is something you will encounter a lot in your life as a data practitioner. A CSV is a comma-separated file where data from a tabilar format is generally stored and separated using commas, although other characters can also be used.

In this activity, we will be tasked with building our own CSV reader and parser. Although it's a big task if we try to cover all use cases and edge cases, along with escape characters and all, for the sake of this small activity, we will keep our requirements small. We will assume that there is no escape character, meaning that if you use a comma at any place in your row, it means you are

starting a new column. We will also assume that the only function we are interested in is to be able to read a CSV file line by line where each read will generate a new dict with the column names as keys and row anmes as values.

Here is an example.

Name	Age	Location
Bob	24	California

We can convert the data in the preceding table into a Python dictionary, which would look as follows:

[32]:
```
our_dict = {"Name":"Bob", "Age":"24", "Location":"California"}
```

1) Import `zip_longest` from itertools. Create a function to zip `header`, `line` and `fillvalue=None`

[33]:
```
from itertools import zip_longest
```

[34]:
```
def zipper(header, line):
    zipped_line = zip_longest(header, line, fillvalue=None)
    return_dict = dict()
    for key, value in zipped_line:
        return_dict[key] = value
    return return_dict
```

2) Open the accompanying `sales_record.csv` file from the Github link by using `r` mode inside a with block and first check that it is opened

[35]:
```
with open("sales_record.csv", "r") as file:
    for i, line in enumerate(file):
        print(line)
        if i > 10:
            break
```

```
Region,Country,Item Type,Sales Channel,Order Priority,Order Date,Order ID,Ship
Date,Units Sold,Unit Price,Unit Cost,Total Revenue,Total Cost,Total Profit

Central America and the Caribbean,Antigua and Barbuda ,Baby Food,Online,M,12/20/
2013,957081544,1/11/2014,552,255.28,159.42,140914.56,87999.84,52914.72

Central America and the Caribbean,Panama,Snacks,Offline,C,7/5/2010,301644504,7/2
6/2010,2167,152.58,97.44,330640.86,211152.48,119488.38

Europe,Czech Republic,Beverages,Offline,C,9/12/2011,478051030,9/29/2011,4778,47.
45,31.79,226716.10,151892.62,74823.48

Asia,North Korea,Cereal,Offline,L,5/13/2010,892599952,6/15/2010,9016,205.70,117.
11,1854591.20,1055863.76,798727.44
```

```
Asia,Sri Lanka,Snacks,Offline,C,7/20/2015,571902596,7/27/2015,7542,152.58,97.44,
1150758.36,734892.48,415865.88

Middle East and North Africa,Morocco,Personal Care,Offline,L,11/8/2010,412882792
,11/22/2010,48,81.73,56.67,3923.04,2720.16,1202.88

Australia and Oceania,Federated States of Micronesia,Clothes,Offline,H,3/28/2011
,932776868,5/10/2011,8258,109.28,35.84,902434.24,295966.72,606467.52

Europe,Bosnia and Herzegovina,Clothes,Online,M,10/14/2013,919133651,11/4/2013,92
7,109.28,35.84,101302.56,33223.68,68078.88

Middle East and North Africa,Afghanistan,Clothes,Offline,M,8/27/2016,579814469,1
0/5/2016,8841,109.28,35.84,966144.48,316861.44,649283.04

Sub-Saharan Africa,Ethiopia,Baby Food,Online,M,4/13/2015,192993152,5/7/2015,9817
,255.28,159.42,2506083.76,1565026.14,941057.62

Middle East and North Africa,Turkey,Office Supplies,Offline,C,9/25/2013,55715602
6,10/15/2013,3704,651.21,524.96,2412081.84,1944451.84,467630.00
```

3) Read the first line and use string methods to generate a list of all column names

[36]:
```
with open("sales_record.csv", "r") as file:
    column_names = file.readline()
    column_names = column_names.replace("\n", "").split(",")
    print(column_names)
```

```
['Region', 'Country', 'Item Type', 'Sales Channel', 'Order Priority', 'Order
Date', 'Order ID', 'Ship Date', 'Units Sold', 'Unit Price', 'Unit Cost', 'Total
Revenue', 'Total Cost', 'Total Profit']
```

4) Start reading the file. Read it line by line.

[37]:
```
display(HTML("<style>.container { width:210% !important; }</style>"))
```

```
<IPython.core.display.HTML object>
```

[38]:
```
with open("sales_record.csv", "r") as file:
    column_names = file.readline()
    column_names = column_names.replace("\n", "").split(",")
    print(str(len(column_names)*" {:<35} |").format(*column_names))
    print(len((str(len(column_names)*" {:^35} |").format(*column_names)))*"-")
    for i, line in enumerate(file):
        print(str(len(column_names)*" {:<35} |").format(*line.replace("\n", "").
↪split(",")))
        if i > 20:
```

```
            break
```

```
 Region                                | Country                                |
Item Type                             | Sales Channel                          |
Order Priority                        | Order Date                             |
Order ID                              | Ship Date                              |
Units Sold                            | Unit Price                             | Unit
Cost                                | Total Revenue                          | Total
Cost                               | Total Profit                           |
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
--------------------------------------------------
 Central America and the Caribbean    | Antigua and Barbuda                    |
Baby Food                             | Online                                 | M
| 12/20/2013                           | 957081544                              |
1/11/2014                             | 552                                    |
255.28                                | 159.42                                 |
140914.56                             | 87999.84                               |
52914.72                              |
 Central America and the Caribbean    | Panama                                 |
Snacks                                | Offline                                | C
| 7/5/2010                            | 301644504                              |
7/26/2010                             | 2167                                   |
152.58                                | 97.44                                  |
330640.86                             | 211152.48                              |
119488.38                             |
 Europe                               | Czech Republic                         |
Beverages                             | Offline                                | C
| 9/12/2011                           | 478051030                              |
9/29/2011                             | 4778                                   |
47.45                                 | 31.79                                  |
226716.10                             | 151892.62                              |
74823.48                              |
 Asia                                 | North Korea                            |
Cereal                                | Offline                                | L
| 5/13/2010                           | 892599952                              |
6/15/2010                             | 9016                                   |
205.70                                | 117.11                                 |
1854591.20                            | 1055863.76                             |
798727.44                             |
 Asia                                 | Sri Lanka                              |
Snacks                                | Offline                                | C
| 7/20/2015                           | 571902596                              |
```

7/27/2015 | 7542 |
152.58 | 97.44 |
1150758.36 | 734892.48 |
415865.88 |
Middle East and North Africa | Morocco |
Personal Care | Offline | L
| 11/8/2010 | 412882792 |
11/22/2010 | 48 |
81.73 | 56.67 |
3923.04 | 2720.16 |
1202.88 |
Australia and Oceania | Federated States of Micronesia |
Clothes | Offline | H
| 3/28/2011 | 932776868 |
5/10/2011 | 8258 |
109.28 | 35.84 |
902434.24 | 295966.72 |
606467.52 |
Europe | Bosnia and Herzegovina |
Clothes | Online | M
| 10/14/2013 | 919133651 |
11/4/2013 | 927 |
109.28 | 35.84 |
101302.56 | 33223.68 |
68078.88 |
Middle East and North Africa | Afghanistan |
Clothes | Offline | M
| 8/27/2016 | 579814469 |
10/5/2016 | 8841 |
109.28 | 35.84 |
966144.48 | 316861.44 |
649283.04 |
Sub-Saharan Africa | Ethiopia |
Baby Food | Online | M
| 4/13/2015 | 192993152 |
5/7/2015 | 9817 |
255.28 | 159.42 |
2506083.76 | 1565026.14 |
941057.62 |
Middle East and North Africa | Turkey |
Office Supplies | Offline | C
| 9/25/2013 | 557156026 |
10/15/2013 | 3704 |
651.21 | 524.96 |
2412081.84 | 1944451.84 |
467630.00 |
Middle East and North Africa | Oman |
Cosmetics | Online | M

| 5/12/2013 | 741101920 |
5/17/2013 | 7382 |
437.20 | 263.33 |
3227410.40 | 1943902.06 |
1283508.34 |
 Asia | Malaysia |
Cereal | Offline | L
| 7/31/2016 | 333942162 |
8/25/2016 | 9762 |
205.70 | 117.11 |
2008043.40 | 1143227.82 |
864815.58 |
 Central America and the Caribbean | Saint Lucia |
Cosmetics | Offline | H
| 7/6/2015 | 795100581 |
7/16/2015 | 6786 |
437.20 | 263.33 |
2966839.20 | 1786957.38 |
1179881.82 |
 Central America and the Caribbean | Saint Vincent and the Grenadines |
Baby Food | Online | L
| 11/28/2010 | 504313504 |
12/3/2010 | 6428 |
255.28 | 159.42 |
1640939.84 | 1024751.76 |
616188.08 |
 Middle East and North Africa | Lebanon |
Meat | Offline | H
| 12/17/2015 | 611629760 |
1/31/2016 | 3693 |
421.89 | 364.69 |
1558039.77 | 1346800.17 |
211239.60 |
 Europe | Austria |
Cereal | Offline | C
| 8/13/2014 | 987410676 |
9/6/2014 | 5616 |
205.70 | 117.11 |
1155211.20 | 657689.76 |
497521.44 |
 Europe | Bulgaria |
Office Supplies | Online | L
| 10/31/2010 | 672330081 |
11/29/2010 | 6266 |
651.21 | 524.96 |
4080481.86 | 3289399.36 |
791082.50 |
 North America | Mexico |

```
Beverages                              | Online                                | C
| 3/13/2017                             | 127374303                             |
3/20/2017                              | 1742                                  |
47.45                                  | 31.79                                 |
82657.90                               | 55378.18                              |
27279.72                               |
 Central America and the Caribbean     | Trinidad and Tobago                   |
Baby Food                              | Offline                               | C
| 4/16/2013                             | 783842170                             |
6/1/2013                               | 5172                                  |
255.28                                 | 159.42                                |
1320308.16                             | 824520.24                             |
495787.92                              |
 Middle East and North Africa          | Libya                                 |
Beverages                              | Offline                               | L
| 1/18/2010                             | 993345010                             |
3/3/2010                               | 1718                                  |
47.45                                  | 31.79                                 |
81519.10                               | 54615.22                              |
26903.88                               |
 Middle East and North Africa          | Algeria                               |
Baby Food                              | Offline                               | M
| 9/5/2015                              | 977806651                             |
10/14/2015                             | 3572                                  |
255.28                                 | 159.42                                |
911860.16                              | 569448.24                             |
342411.92                              |
```

[39]:
```
# display(HTML("<style>.container { width:56% !important; }</style>"))
```

5) Read each line and pass that line to a function along with the list of headers. The work of the function is to construct a dict out of these two and fill up the key/values. Keep in ind that a missing value should results in `None`

[40]:
```
with open("sales_record.csv", "r") as file:
    dict_list = []
    column_names = file.readline()
    column_names = column_names.replace("\n", "").split(",")
    for i, line in enumerate(file):
        x = line.replace("\n", "").split(",")
        my_dict = zipper(column_names, x)
        dict_list.append(my_dict)
        if i > 20:
            break
print(dict_list)
```

```
[{'Region': 'Central America and the Caribbean', 'Country': 'Antigua and Barbuda
', 'Item Type': 'Baby Food', 'Sales Channel': 'Online', 'Order Priority': 'M',
'Order Date': '12/20/2013', 'Order ID': '957081544', 'Ship Date': '1/11/2014',
```

'Units Sold': '552', 'Unit Price': '255.28', 'Unit Cost': '159.42', 'Total Revenue': '140914.56', 'Total Cost': '87999.84', 'Total Profit': '52914.72'}, {'Region': 'Central America and the Caribbean', 'Country': 'Panama', 'Item Type': 'Snacks', 'Sales Channel': 'Offline', 'Order Priority': 'C', 'Order Date': '7/5/2010', 'Order ID': '301644504', 'Ship Date': '7/26/2010', 'Units Sold': '2167', 'Unit Price': '152.58', 'Unit Cost': '97.44', 'Total Revenue': '330640.86', 'Total Cost': '211152.48', 'Total Profit': '119488.38'}, {'Region': 'Europe', 'Country': 'Czech Republic', 'Item Type': 'Beverages', 'Sales Channel': 'Offline', 'Order Priority': 'C', 'Order Date': '9/12/2011', 'Order ID': '478051030', 'Ship Date': '9/29/2011', 'Units Sold': '4778', 'Unit Price': '47.45', 'Unit Cost': '31.79', 'Total Revenue': '226716.10', 'Total Cost': '151892.62', 'Total Profit': '74823.48'}, {'Region': 'Asia', 'Country': 'North Korea', 'Item Type': 'Cereal', 'Sales Channel': 'Offline', 'Order Priority': 'L', 'Order Date': '5/13/2010', 'Order ID': '892599952', 'Ship Date': '6/15/2010', 'Units Sold': '9016', 'Unit Price': '205.70', 'Unit Cost': '117.11', 'Total Revenue': '1854591.20', 'Total Cost': '1055863.76', 'Total Profit': '798727.44'}, {'Region': 'Asia', 'Country': 'Sri Lanka', 'Item Type': 'Snacks', 'Sales Channel': 'Offline', 'Order Priority': 'C', 'Order Date': '7/20/2015', 'Order ID': '571902596', 'Ship Date': '7/27/2015', 'Units Sold': '7542', 'Unit Price': '152.58', 'Unit Cost': '97.44', 'Total Revenue': '1150758.36', 'Total Cost': '734892.48', 'Total Profit': '415865.88'}, {'Region': 'Middle East and North Africa', 'Country': 'Morocco', 'Item Type': 'Personal Care', 'Sales Channel': 'Offline', 'Order Priority': 'L', 'Order Date': '11/8/2010', 'Order ID': '412882792', 'Ship Date': '11/22/2010', 'Units Sold': '48', 'Unit Price': '81.73', 'Unit Cost': '56.67', 'Total Revenue': '3923.04', 'Total Cost': '2720.16', 'Total Profit': '1202.88'}, {'Region': 'Australia and Oceania', 'Country': 'Federated States of Micronesia', 'Item Type': 'Clothes', 'Sales Channel': 'Offline', 'Order Priority': 'H', 'Order Date': '3/28/2011', 'Order ID': '932776868', 'Ship Date': '5/10/2011', 'Units Sold': '8258', 'Unit Price': '109.28', 'Unit Cost': '35.84', 'Total Revenue': '902434.24', 'Total Cost': '295966.72', 'Total Profit': '606467.52'}, {'Region': 'Europe', 'Country': 'Bosnia and Herzegovina', 'Item Type': 'Clothes', 'Sales Channel': 'Online', 'Order Priority': 'M', 'Order Date': '10/14/2013', 'Order ID': '919133651', 'Ship Date': '11/4/2013', 'Units Sold': '927', 'Unit Price': '109.28', 'Unit Cost': '35.84', 'Total Revenue': '101302.56', 'Total Cost': '33223.68', 'Total Profit': '68078.88'}, {'Region': 'Middle East and North Africa', 'Country': 'Afghanistan', 'Item Type': 'Clothes', 'Sales Channel': 'Offline', 'Order Priority': 'M', 'Order Date': '8/27/2016', 'Order ID': '579814469', 'Ship Date': '10/5/2016', 'Units Sold': '8841', 'Unit Price': '109.28', 'Unit Cost': '35.84', 'Total Revenue': '966144.48', 'Total Cost': '316861.44', 'Total Profit': '649283.04'}, {'Region': 'Sub-Saharan Africa', 'Country': 'Ethiopia', 'Item Type': 'Baby Food', 'Sales Channel': 'Online', 'Order Priority': 'M', 'Order Date': '4/13/2015', 'Order ID': '192993152', 'Ship Date': '5/7/2015', 'Units Sold': '9817', 'Unit Price': '255.28', 'Unit Cost': '159.42', 'Total Revenue': '2506083.76', 'Total Cost': '1565026.14', 'Total Profit': '941057.62'}, {'Region': 'Middle East and North Africa', 'Country': 'Turkey', 'Item Type': 'Office Supplies', 'Sales Channel': 'Offline', 'Order Priority': 'C', 'Order Date': '9/25/2013', 'Order ID': '557156026', 'Ship Date':

'10/15/2013', 'Units Sold': '3704', 'Unit Price': '651.21', 'Unit Cost':
'524.96', 'Total Revenue': '2412081.84', 'Total Cost': '1944451.84', 'Total
Profit': '467630.00'}, {'Region': 'Middle East and North Africa', 'Country':
'Oman', 'Item Type': 'Cosmetics', 'Sales Channel': 'Online', 'Order Priority':
'M', 'Order Date': '5/12/2013', 'Order ID': '741101920', 'Ship Date':
'5/17/2013', 'Units Sold': '7382', 'Unit Price': '437.20', 'Unit Cost':
'263.33', 'Total Revenue': '3227410.40', 'Total Cost': '1943902.06', 'Total
Profit': '1283508.34'}, {'Region': 'Asia', 'Country': 'Malaysia', 'Item Type':
'Cereal', 'Sales Channel': 'Offline', 'Order Priority': 'L', 'Order Date':
'7/31/2016', 'Order ID': '333942162', 'Ship Date': '8/25/2016', 'Units Sold':
'9762', 'Unit Price': '205.70', 'Unit Cost': '117.11', 'Total Revenue':
'2008043.40', 'Total Cost': '1143227.82', 'Total Profit': '864815.58'},
{'Region': 'Central America and the Caribbean', 'Country': 'Saint Lucia', 'Item
Type': 'Cosmetics', 'Sales Channel': 'Offline', 'Order Priority': 'H', 'Order
Date': '7/6/2015', 'Order ID': '795100581', 'Ship Date': '7/16/2015', 'Units
Sold': '6786', 'Unit Price': '437.20', 'Unit Cost': '263.33', 'Total Revenue':
'2966839.20', 'Total Cost': '1786957.38', 'Total Profit': '1179881.82'},
{'Region': 'Central America and the Caribbean', 'Country': 'Saint Vincent and
the Grenadines', 'Item Type': 'Baby Food', 'Sales Channel': 'Online', 'Order
Priority': 'L', 'Order Date': '11/28/2010', 'Order ID': '504313504', 'Ship
Date': '12/3/2010', 'Units Sold': '6428', 'Unit Price': '255.28', 'Unit Cost':
'159.42', 'Total Revenue': '1640939.84', 'Total Cost': '1024751.76', 'Total
Profit': '616188.08'}, {'Region': 'Middle East and North Africa', 'Country':
'Lebanon', 'Item Type': 'Meat', 'Sales Channel': 'Offline', 'Order Priority':
'H', 'Order Date': '12/17/2015', 'Order ID': '611629760', 'Ship Date':
'1/31/2016', 'Units Sold': '3693', 'Unit Price': '421.89', 'Unit Cost':
'364.69', 'Total Revenue': '1558039.77', 'Total Cost': '1346800.17', 'Total
Profit': '211239.60'}, {'Region': 'Europe', 'Country': 'Austria', 'Item Type':
'Cereal', 'Sales Channel': 'Offline', 'Order Priority': 'C', 'Order Date':
'8/13/2014', 'Order ID': '987410676', 'Ship Date': '9/6/2014', 'Units Sold':
'5616', 'Unit Price': '205.70', 'Unit Cost': '117.11', 'Total Revenue':
'1155211.20', 'Total Cost': '657689.76', 'Total Profit': '497521.44'},
{'Region': 'Europe', 'Country': 'Bulgaria', 'Item Type': 'Office Supplies',
'Sales Channel': 'Online', 'Order Priority': 'L', 'Order Date': '10/31/2010',
'Order ID': '672330081', 'Ship Date': '11/29/2010', 'Units Sold': '6266', 'Unit
Price': '651.21', 'Unit Cost': '524.96', 'Total Revenue': '4080481.86', 'Total
Cost': '3289399.36', 'Total Profit': '791082.50'}, {'Region': 'North America',
'Country': 'Mexico', 'Item Type': 'Beverages', 'Sales Channel': 'Online', 'Order
Priority': 'C', 'Order Date': '3/13/2017', 'Order ID': '127374303', 'Ship Date':
'3/20/2017', 'Units Sold': '1742', 'Unit Price': '47.45', 'Unit Cost': '31.79',
'Total Revenue': '82657.90', 'Total Cost': '55378.18', 'Total Profit':
'27279.72'}, {'Region': 'Central America and the Caribbean', 'Country':
'Trinidad and Tobago', 'Item Type': 'Baby Food', 'Sales Channel': 'Offline',
'Order Priority': 'C', 'Order Date': '4/16/2013', 'Order ID': '783842170', 'Ship
Date': '6/1/2013', 'Units Sold': '5172', 'Unit Price': '255.28', 'Unit Cost':
'159.42', 'Total Revenue': '1320308.16', 'Total Cost': '824520.24', 'Total
Profit': '495787.92'}, {'Region': 'Middle East and North Africa', 'Country':
'Libya', 'Item Type': 'Beverages', 'Sales Channel': 'Offline', 'Order Priority':

```
'L', 'Order Date': '1/18/2010', 'Order ID': '993345010', 'Ship Date':
'3/3/2010', 'Units Sold': '1718', 'Unit Price': '47.45', 'Unit Cost': '31.79',
'Total Revenue': '81519.10', 'Total Cost': '54615.22', 'Total Profit':
'26903.88'}, {'Region': 'Middle East and North Africa', 'Country': 'Algeria',
'Item Type': 'Baby Food', 'Sales Channel': 'Offline', 'Order Priority': 'M',
'Order Date': '9/5/2015', 'Order ID': '977806651', 'Ship Date': '10/14/2015',
'Units Sold': '3572', 'Unit Price': '255.28', 'Unit Cost': '159.42', 'Total
Revenue': '911860.16', 'Total Cost': '569448.24', 'Total Profit': '342411.92'}]
```

[41]:
```
# Sanity check to prove that my list can be interpreted as a dataframe

pd.DataFrame(dict_list)
```

[41]:

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID
0	Central America and the Caribbean	Antigua and Barbuda	Baby Food	Online	M	12/20/2013	957081544
1	Central America and the Caribbean	Panama	Snacks	Offline	C	7/5/2010	301644504
2	Europe	Czech Republic	Beverages	Offline	C	9/12/2011	478051030
3	Asia	North Korea	Cereal	Offline	L	5/13/2010	892599952
4	Asia	Sri Lanka	Snacks	Offline	C	7/20/2015	571902596
5	Middle East and North Africa	Morocco	Personal Care	Offline	L	11/8/2010	412882792
6	Australia and Oceania	Federated States of Micronesia	Clothes	Offline	H	3/28/2011	932776868
7	Europe	Bosnia and Herzegovina	Clothes	Online	M	10/14/2013	919133651
8	Middle East and North Africa	Afghanistan	Clothes	Offline	M	8/27/2016	579814469
9	Sub-Saharan Africa	Ethiopia					
10	Middle East and North Africa	Turkey					
11	Middle East and North Africa	Oman					
12	Asia	Malaysia					
13	Central America and the Caribbean	Saint Lucia					
14	Central America and the Caribbean	Saint Vincent and the Grenadines					
15	Middle East and North Africa	Lebanon					
16	Europe	Austria					
17	Europe	Bulgaria					
18	North America	Mexico					
19	Central America and the Caribbean	Trinidad and Tobago					
20	Middle East and North Africa	Libya					
21	Middle East and North Africa	Algeria					

9	Baby Food	Online	M	4/13/2015	192993152
10	Office Supplies	Offline	C	9/25/2013	557156026
11	Cosmetics	Online	M	5/12/2013	741101920
12	Cereal	Offline	L	7/31/2016	333942162
13	Cosmetics	Offline	H	7/6/2015	795100581
14	Baby Food	Online	L	11/28/2010	504313504
15	Meat	Offline	H	12/17/2015	611629760
16	Cereal	Offline	C	8/13/2014	987410676
17	Office Supplies	Online	L	10/31/2010	672330081
18	Beverages	Online	C	3/13/2017	127374303
19	Baby Food	Offline	C	4/16/2013	783842170
20	Beverages	Offline	L	1/18/2010	993345010
21	Baby Food	Offline	M	9/5/2015	977806651

	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	\
0	1/11/2014	552	255.28	159.42	140914.56	87999.84	
1	7/26/2010	2167	152.58	97.44	330640.86	211152.48	
2	9/29/2011	4778	47.45	31.79	226716.10	151892.62	
3	6/15/2010	9016	205.70	117.11	1854591.20	1055863.76	
4	7/27/2015	7542	152.58	97.44	1150758.36	734892.48	
5	11/22/2010	48	81.73	56.67	3923.04	2720.16	
6	5/10/2011	8258	109.28	35.84	902434.24	295966.72	
7	11/4/2013	927	109.28	35.84	101302.56	33223.68	
8	10/5/2016	8841	109.28	35.84	966144.48	316861.44	
9	5/7/2015	9817	255.28	159.42	2506083.76	1565026.14	
10	10/15/2013	3704	651.21	524.96	2412081.84	1944451.84	
11	5/17/2013	7382	437.20	263.33	3227410.40	1943902.06	
12	8/25/2016	9762	205.70	117.11	2008043.40	1143227.82	
13	7/16/2015	6786	437.20	263.33	2966839.20	1786957.38	
14	12/3/2010	6428	255.28	159.42	1640939.84	1024751.76	
15	1/31/2016	3693	421.89	364.69	1558039.77	1346800.17	
16	9/6/2014	5616	205.70	117.11	1155211.20	657689.76	
17	11/29/2010	6266	651.21	524.96	4080481.86	3289399.36	
18	3/20/2017	1742	47.45	31.79	82657.90	55378.18	
19	6/1/2013	5172	255.28	159.42	1320308.16	824520.24	
20	3/3/2010	1718	47.45	31.79	81519.10	54615.22	
21	10/14/2015	3572	255.28	159.42	911860.16	569448.24	

	Total Profit
0	52914.72
1	119488.38
2	74823.48
3	798727.44
4	415865.88
5	1202.88
6	606467.52
7	68078.88

```
8     649283.04
9     941057.62
10    467630.00
11   1283508.34
12    864815.58
13   1179881.82
14    616188.08
15    211239.60
16    497521.44
17    791082.50
18     27279.72
19    495787.92
20     26903.88
21    342411.92
```

[ ]: