# Lab 1: D7054E

Sebastian Morel (sebmor-8)
Patrick Pilipiec (patpil-9)

**Group 7**

February 11, 2021

## Introduction

Lab 1 provides us with two sets of data. Data set 1 is published by the Sustainable Development Solutions Network and showcases a number of happiness factors by country [1]. Data set 2 is from the World Health Organisation, and presents COVID-19 statistics by country [2]. Lab 1 is divided up into two parts part A and part B. In part A, the goal is to work with the two provided data sets and create data structures, calculate the mean, variance and standard deviation, and find the min and max values for each column. This needs to be achieved by utilizing object oriented concepts like inheritance and polymorphism. In part B the aim is getting the data and visualize different aspects of it, through line charts, bar charts, heatmaps and scatterplots, additionally the data will be sourced by different ways 1) API 2) CSV 3) webscraping 4) mongoDB.

## 1 Methodology

The lab was performed in Spyder inside a virtual environment using the Anaconda navigator, in order to isolate things such as library versions from other virtual environments and on the system. The data sets used in this lab were provided by the World Health Organisation for the COVID-19 statistics, and the Sustainable Development Solutions Network for the World Happiness Report Dataset.

### 1.1 Techniques

For Part A, we aimed to use only vanilla Python and the packages that come by default. Indeed, we could have used Pandas and other packages for manipulating data, but we found that this approach may be considered ”cheating”. Our objective was write the required functionality from scratch. In addition to Python 3.x, we used the csv and math packages.

For Part B, we used pandas[3] to convert the csv file to a dataframe and to more seamlessly preprocess the data for the plotting libraries. To create the plots for part B we used pyplot[4] from matplotlib, which is very easy to use in order to create simple plots, furthermore it is well integrated with pandas. The heatmap was created through seaborn[5] which is a data visualization library based on matplotlib and is more intuitive to use when it comes to arrays and mapping them to a heatmap than pyplot. For the database we utilized mongodb, and thus used the pymongo python library. To webscrape we used BeatifulSoup to process the html, and selenium to get it because of the data being dynamic content on the webscraped website.

### 1.2 The dataset

The two provided datasets were:

1. ”WHO COVID-19 data” dataset. This dataset contains statistics about the COVID-19 infection for 238 countries. The dataset contains the following 11 columns.

Table 1. WHO COVID-19 data type and name for each column.

| Data type | Column name |
|---|---|
| str | Name |
| str | WHO Region |
| int | Cases - cumulative total |
| float | Cases - cumulative total per 1 million population |
| int | Cases - newly reported in last 7 days |
| int | Cases - newly reported in last 24 hours |
| int | Deaths - cumulative total |
| float | Deaths - cumulative total per 1 million population |
| int | Deaths - newly reported in last 7 days |
| int | Deaths - newly reported in last 24 hours |
| str | Transmission Classification |

2. ”World Happiness 2019” dataset. This dataset ranks 156 countries using a happiness score, and provides numerical details on various variables. It contains the following nine columns.

Table 2. World Happiness 2019 data type and name for each column.

| Data type | Column name |
|---|---|
| int | Overall rank |
| str | Country or region |
| float | Score |
| float | GDP per capita |
| float | Social support |
| float | Healthy life expectancy |
| float | Freedom to make life choices |
| float | Generosity |
| float | Perceptions of corruption |

### 1.3 The pre-processing of the dataset

Part A, didn't utilize any pre-processing.
Part B we used numerous techniques to pre-process the datasets. We formated the data according to the instructions, depending on what we wanted to plot. For the bar chart we got the data from our mongodb database and converted it directly to a dataframe, we then used pandas feature ”groupby()” to group all 'Cases - cumulative total' and 'Deaths - cumulative total' into their respective WHO Region and taking the mean value off them to use in the plot. For the line chart, we got the data by webscraping the WHO (World Health Organisation) website for COVID-19 data[6]. The WHO website's corona data was dynamic content therefore we used selenium on top of beautifulsoup to be able to fetch it. We then extracted the 'Country' and 'Cumulative cases' columns from the website and created two lists out of them, and plotted them. For the heatmap we converted the csv file to a pandas dataframe. When plotting the heatmap we chose to only use a number of country which square root would result in an integer, to make an evenly squared heatmap this resulted in us removing 11 countries who hade a 'Deaths per million' at 0. The first preprocessing technique used for the heatmap after converting it to a dataframe was to convert the list into a numpy array in order to reshape it into a 15x15 numpy array in order for seaborn to be able to process it to create a heatmap. To plot the scatter plot, we used a covid19 API[7] and converted it to a dataframe, we picked out the ”Cumulative Cases” and ”Cumulative deaths” columns and created two lists and plotted them against eachother.

## 2 Results

### 2.1 Part A

This problem was approached by creating an abstraction for a DataFrame and a class for each of the two data files, namely Covid and WorldHappiness2019. The classes Covid and WorldHappiness2019 inherit from the class DataFrame. See the UML-diagram in Appendix A.
All the functionalities for reading and pre-processing data, as well as for displaying information and statistics about these datasets, was implemented in the class DataFrame. Encapsulation was applied by limiting the scope of class variables and methods, such that those for internal purposes cannot be accessed from outside the class. The public methods, such as ”print_columns()”, ”columns()”, and ”info()”, are intended to be publicly accessible by the user.
The two classes representing each file, Covid and WorldHappiness2019, inherit from the class DataFrame and utilize the constructor method to pass information specific to these datasets to their parent class. This information is the

name of the dataset, the path of the dataset, and the data types of its columns.
We refrained from using functionality from Pandas to infer data types because we find that this may defeat the purpose of this assignment, namely to work solely with vanilla Python.
The public "info()" method can be called to display information about the dataset, such as its name, the number of observations in the dataset and the column names. In addition, the public method "describe_columns()" can be called to describe all columns in the dataset. For each column, based on its datatype, various statistics are printed. In particular, for integers and floats, the number of observations, minimum and maximum values, mean and variance, and the standard deviation of each column in computed. For boolean columns, the number of observations, and the number of true and false values are computed. For string values, the number of observations and the count of unique categories are presented. The public method "describe_column()" does the same as "describe_columns()", but only for one column. Lastly, upon running the Python file, the program is configured to test its accuracy and to present information about both datasets by default.

## 2.2 Part B

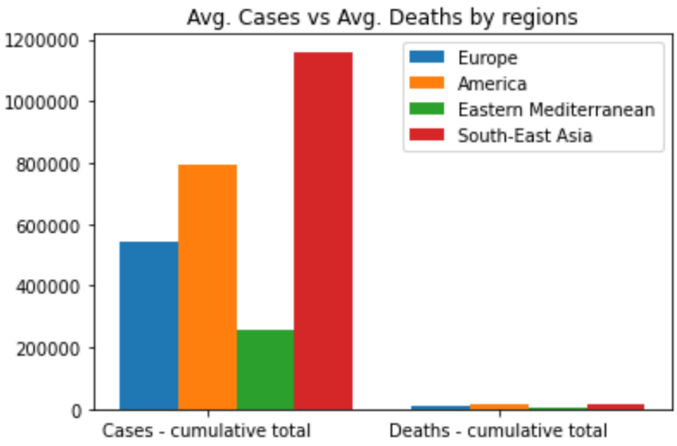


Figure 1: Bar graph comparing cases and deaths in the WHO regions Europe, America, Eastern Mediterranean and South-East Asia.

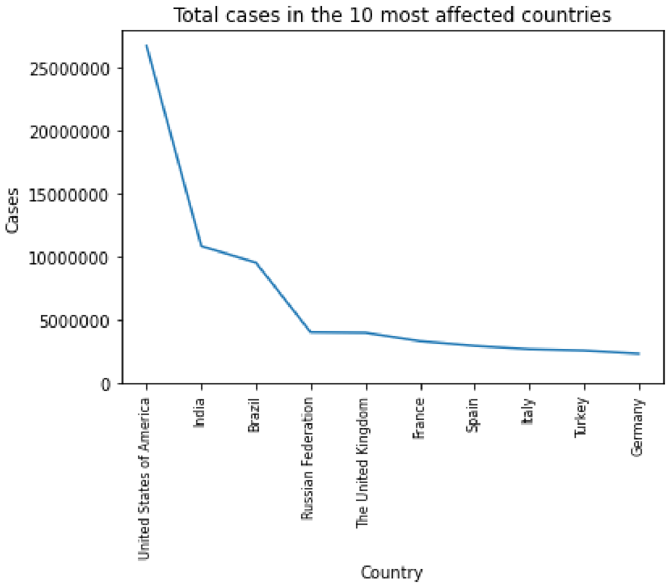


Figure 2: Line chart displaying the total number of deaths in the 10 most affected countries.

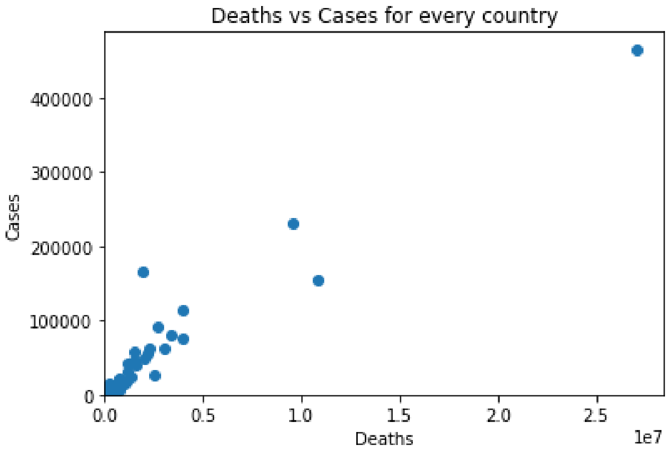


Figure 3: Scatter plot representing the values of cases vs deaths, each dot represents a country.

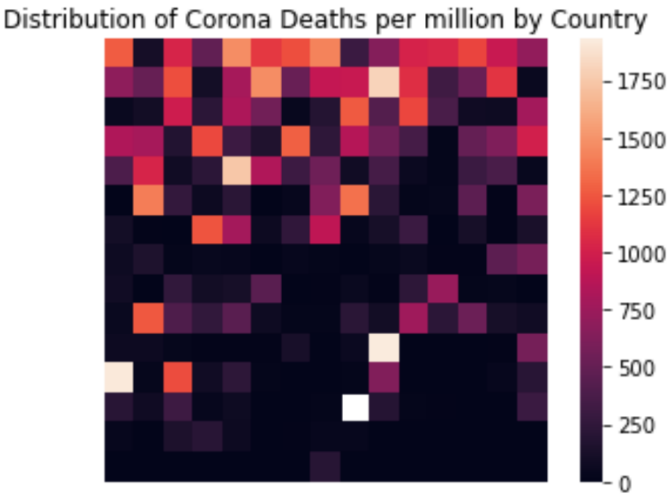


Figure 4: Heatmap showing the distribution of COVID-19 deaths per million each square representing a country, the lighter the square the more deaths per million.

# 3 Conclusion

## 3.1 Part A

The UML-diagram in Appendix A outlines the logic that was created to read and process datasets using good practices in Object-Oriented Programming. The user can easily work with each dataset by instantiating from the class Covid19 or WorldHappiness2019. The created objects give the user access to perform operations on the datasets. This also moves the implementation away to the class DataFrame. Furthermore, the program is very flexible and easy to extend, as the user can easily add a new class for a new file, and inherit all functionality from the parent class DataFrame.

## 3.2 Part B

Figure 1, compares the average cumulative cases for each western regions. The reasoning behind only choosing those regions rather than the whole world is that the selected regions are the ones with the most cases, the amount of cases drops by more than 80% if the next one would be added. That bar would be dwarfed by the other ones.
Figure 2 displays the total amount of deaths in the 10 most affected countries, a bar chart would be preferable to display this kind of information. Optimally we would have used the line chart to display some sort of timeline of daily cases over time, but the provided dataset did not include this. We would've also preferred to group the countries by WHO Region but that data was not available on the webscraped website. In analyzing this chart we can see just how much more cases the U.S. has compared to the rest of the world, and after the third country the slope becomes more stagnant.

The scatter plot in Figure 3 shows the cases versus deaths. In the figure we can see some dots (countries) that are outside the diagonal cluster, if the dots are below it it means that the country has a comparatively high death rate per case, if it is over the cluster it would mean that the country has a comparatively low death rate per case. Despite not knowing which country represents which dot represents which country, the plot gives us a global overview of how countries generally are doing. Here we would've also preferred to use Deaths and Cases per million to get a better view of the cluster, the one dot in the top right is an outlier and makes it harder to see the diagonal cluster, this outlier could be due to is having a much larger population that the rest of the countries and thus more cases and deaths.
The heat map in Figure 4 also gives us a general overlook of how the distribution looks like in the world. From the figure we can see five countries that a particularly lighter than the rest, which would implicate that there are four countries that a particularly performing worse compared to the rest of the world.

## References

[1] "World Happiness Report Dataset.". https://www.kaggle.com/unsdsn/world-happiness?select=2019.csv. Accessed: 2021-01-30.

[2] "WHO Coronavirus Disease (COVID-19) Dashboard.". https://covid19.who.int/table. Accessed: 2021-01-30.

[3] "Python - Data Analysis Library.". https://pandas.pydata.org/. Accessed: 2021-02-02.

[4] "matplotlib.pyplot - Matplotlib 3.3.3 Documentation.". https://matplotlib.org/3.3.3/api/_as_gen/matplotlib.pyplot.html. Accessed: 2021-02-02.

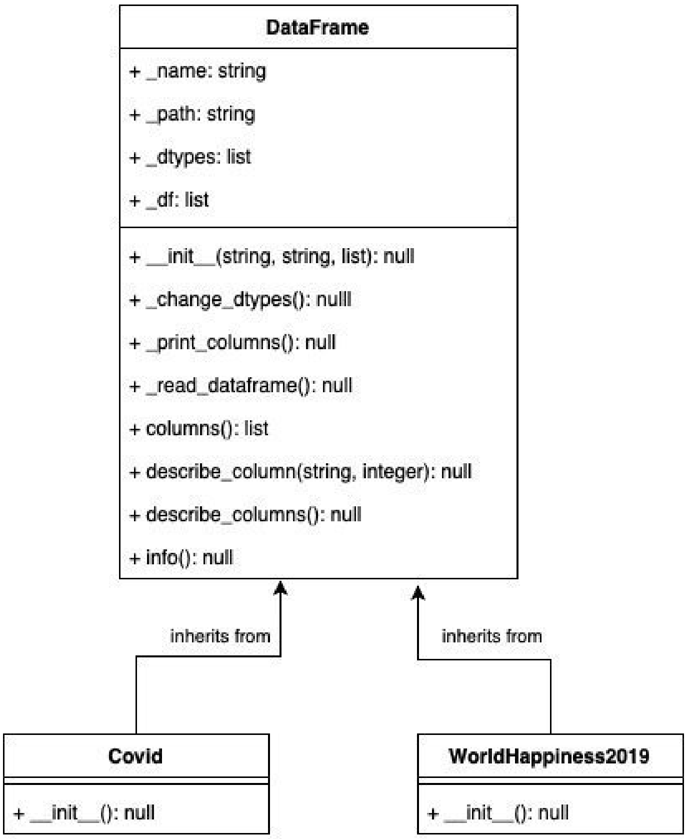
[5] "seaborn: statistical data visualization.". https://seaborn.pydata.org/. Accessed: 2021-02-02.

[6] "Situation by Country, Territory  Area.". https://covid19.who.int/table. Accessed: 2021-02-09.

[7] "COVID-19 API.". https://api.covid19api.com/summary. Accessed: 2021-02-09.

# Appendices

## A UML diagram


DataFrame
+ _name: string
+ _path: string
+ _dtypes: list
+ _df: list
+ __init__(string, string, list): null
+ _change_dtypes(): nulll
+ _print_columns(): null
+ _read_dataframe(): null
+ columns(): list
+ describe_column(string, integer): null
+ describe_columns(): null
+ info(): null
inherits from
inherits from
Covid
+ __init__(): null
WorldHappiness2019
+ __init__(): null