**MANİSA CELAL BAYAR UNIVERSITY**

**CSE3243 (2019-2020) Python Programming**

**TAKE HOME**

( for final exam)

**SPRING 2020**

*ELİF FIRTANA – 170315023*

*4 June 2020*

**Covid-19 Density Reporting**

The first thing we need to do if we want to start plotting and analyzing real world datasets is to get our data. We are gonna be working with the Covid 19 dataset.

We need Covid 19 dataset that maps out cases by counties so we are gonna grab that from kaggle.

Kaggle is a huge online data science community that has a lot of different databases and contents that you can use for analyzing data and working with different libraries.

So we are gonna search for Corona counties;

<https://www.kaggle.com/fireballbyedimyrnmom/us-counties-covid-19-dataset>

<https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties.csv>

It is managed by the New York Times and it's updated.

Before, we install anaconda it comes with spider which we can write our programs in and is also comes with the two libraries that we use matplotlib and pandas.



Here I tried to import two libraries that I need; these are "matplotlib" and "pandas".



*Matplotlib* is a comprehensive library for creating static, animated, and interactive visualizations in Python.

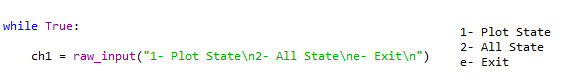
*Pandas* is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

They are imported in and referenced these variables PLT for matplotlib and PD for pandas. So we are gonna stick with that same syntax here now once we have those imported we can import our data.

We are gonna references “pd.read\_csv” because that's the type of file that our data was in and then we can enter the url of our dataset.



We continued with the simple console query;



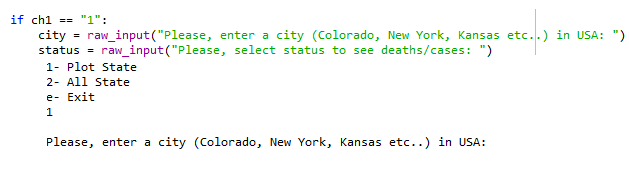
Note: if you are using python3 we need to change it as follows.

raw\_input 🡪 input

If I want to see the plot state cases / deaths of a city, I should choose the first option.

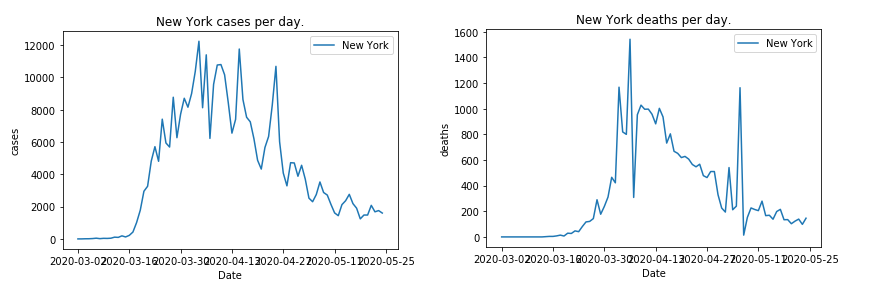
If I want to see the top ten cities with the most cases / deaths on a single chart, I should choose the second option.

If I choose the "Plot State" option, this time we will be asked a question again from the console.



We have the right to choose many of the cities available in our data.

Example; New York, New Jersey, Illinois, California, Massachusetts, Pennsylvania, Texas, Michigan, Florida, Maryland, Georgia, Connecticut, Louisiana, Virginia, Indiana, Ohio…



Cases Deaths

1. **SINGLE STATE - CASES / DEATHS:**

# Create a new data frame of a single state.

df\_city = df[ df['state'] == str(city) ].copy()

# Make sure pandas realizes our date column is a date.

df\_city['date'] = pd.to\_datetime(df\_city['date'])

# Sum up all the cases for city by date.

# Use diff method to compair to previous day to get new cases.

series\_city\_sum = df\_city.groupby('date')[str(status)].sum() 🡪 status : deaths / cases

series\_city\_diff = series\_city\_sum.diff()

plt.plot(series\_city\_diff.index, series\_city\_diff.values, label=city)

# To choose what the x and y axis belong to

plt.xlabel("Date")

plt.ylabel(str(status)) 🡪deaths or cases

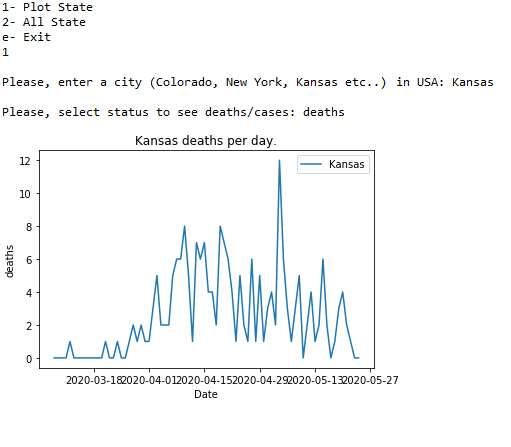
# Graphic title

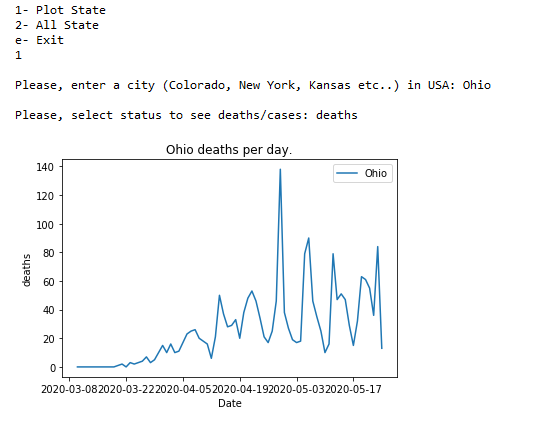
plt.title(str(city)+" "+str(status)+" per day.") 🡪 example : New York Cases per day.

plt.legend()

plt.show()

**Output for Line Plot Deaths / Cases:**





**2. ALL STATE - CASES / DEATHS:**

If we choose the "All State" option, another question will be asked from the console.

ch2 = raw\_input("1- Line Plot\n2- Pie Chart\n")

If you want to see the 10 cities with the most death and incidents on the chart, we should choose the first option.

If we want to see the 5 cities with the most death and incidents on the pie chart, we should choose the second option.

**2.1. ALL STATES IN LINE PLOT**

# Add base style and print out style choices.

plt.style.use('seaborn-colorblind')

print(plt.style.available)

# Get last date to see which states have the most cases currently

last\_date = df['date'].max()

df\_last\_date = df[ df['date'] == last\_date]

series\_last\_date = df\_last\_date.groupby('state')[str(status)].sum()

# I enter a default value here, how many states I want to appear on the chart.

series\_last\_date = series\_last\_date.nlargest(10)

# Remove left and right plot lines.

ax = plt.subplot()

ax.spines["right"].set\_visible(False)

ax.spines["left"].set\_visible(False)

# I set a start date to make it look clearer on the chart

# The chat is flat before 3/15/2020 so lets limit it to after that date.

date\_after = pd.Timestamp("03/15/2020")

for state in series\_last\_date.index:

df\_state = df[ df['state'] == state].copy()

df\_state['date'] = pd.to\_datetime(df\_state['date'])

df\_state = df\_state[ df\_state['date'] > date\_after]

series\_state = df\_state.groupby('date')[str(status)].sum()

series\_state = series\_state.diff()

series\_state.index = series\_state.index.strftime('%b %d')

plt.plot(series\_state.index, series\_state.values, label=state)

plt.xlabel("Date")

plt.ylabel(str(status))

plt.title('Top 10 Covid-19 States')

plt.grid(True)

# Plot custom range for our Y axis

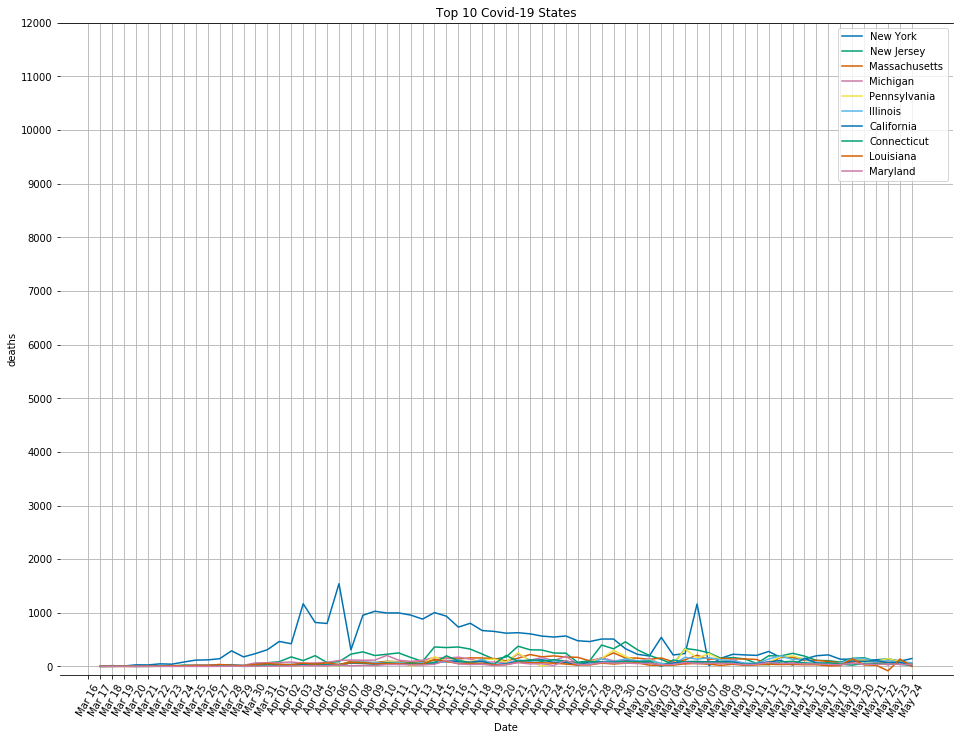
plt.yticks(range(0, 12001, 1000))

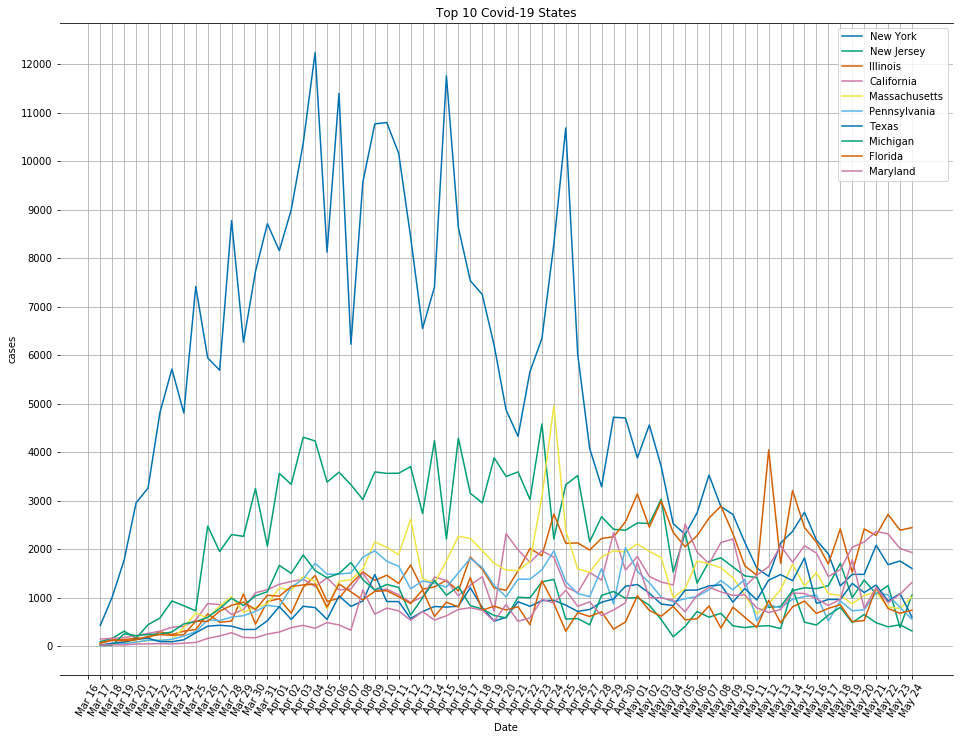
# rotate our dates.

plt.xticks(rotation=60)

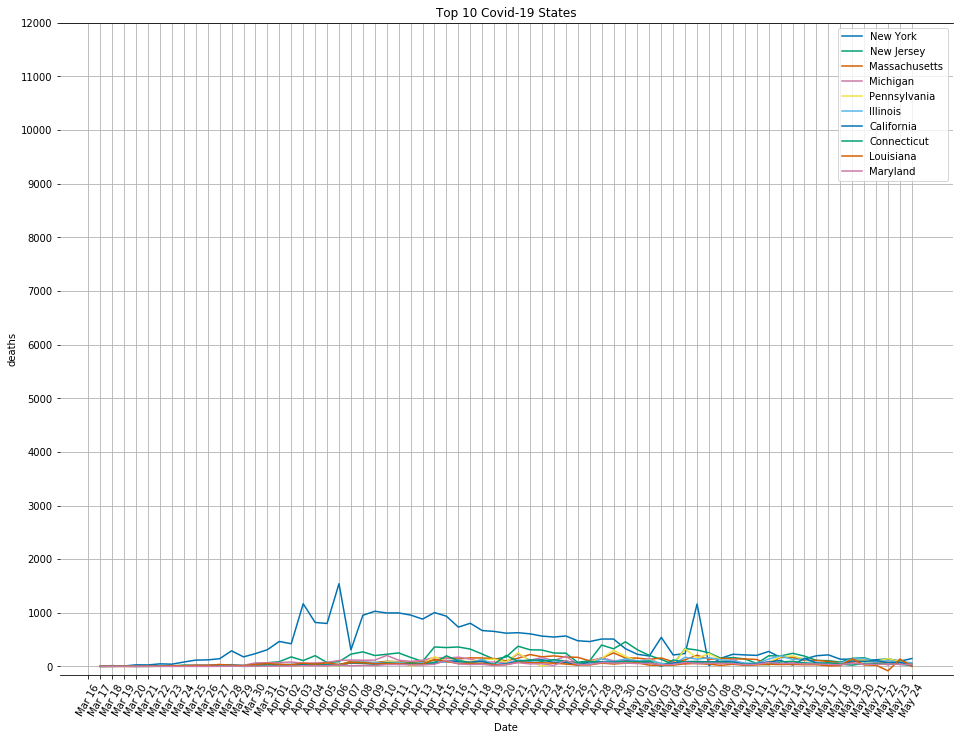
plt.legend()

plt.show() Top 10 States: Deaths





Top 10 States: Cases



US States

* 1. **ALL STATES IN PIE CHART:**

unique\_states = df['state'].unique()

plt.style.use("seaborn-talk")

# Get last date to see which states have the most cases currently

last\_date = df['date'].max()

df\_last\_date = df[ df['date'] == last\_date]

series\_last\_date =df\_last\_date.groupby('state')[str(status)].sum().sort\_values(ascending=False)

print(series\_last\_date)

labels = []

values = []

# how many countries do we want to have on the pie chart, so we have state\_count,

state\_count = 6

other\_total = 0

for state in series\_last\_date.index:

if state\_count > 0:

labels.append(state)

values.append(series\_last\_date[state])

state\_count -= 1

else:

other\_total += series\_last\_date[state]

# We have "other" section for those out of 6 countries

labels.append("Other")

values.append(other\_total)

wedge\_dict = {

'edgecolor': 'white',

'linewidth': 2

}

# There are 6 countries with max values ​​in our pie chart, we should regulate if we want to increase or decrease the number of countries.

# 0.1 gave us the split slice in the shape

explode = (0, 0.1, 0, 0, 0, 0, 0)

plt.pie(values, labels=labels, explode=explode, autopct='%1.1f%%', wedgeprops=wedge\_dict)

plt.show()

* If we want to exit the application, it is enough to write "e".

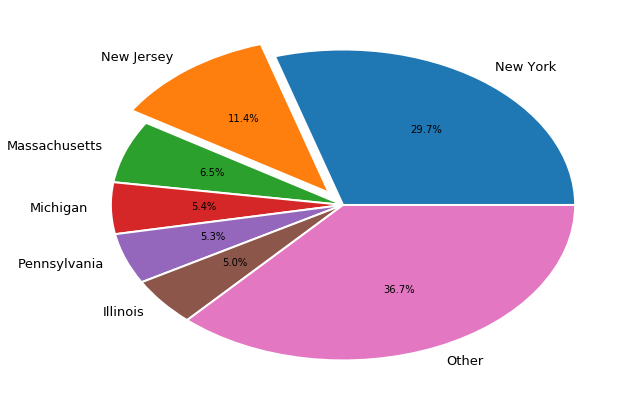
elif ch1 == "e":

print("Exiting...")

break

**Output for Pie Chart:**

*Deaths*



*Cases*

