

Getting Started with Python

Time Series Project

Author :Fredrick Odoyo

Date: 17/3/2022

Resources

1. ##### [Clean Dataset](#)
2. ##### [Submission Portal](#)

If you are having problems please refer to this document:

1. ##### [Time Series Notebook](#)

Instructions

Import all the libraries listed in the first cell. Make sure all modules are installed.

Use the provided data set to answer the following:

1. a) What is the lowest price for Safaricom (*SCOM*) b) What was the date when Safaricom had the lowest price?
2. a) What is the highest price Safaricom stock reached in the data b) What was the date when Safaricom stock recorded the highest price?
3. Create a line plot for Safaricom stock and verify if the information provided above is indeed correct.
4. Select **one** of the sectors provided (agric, comm, bank, const, energy, insur, invest, manu)
 - a) Use **pandas** to create a subset containing all the rows of the dataframe and only companies in your selected sector. Rename this dataframe to the **sector_name_df**
 - b) Using the subset for the sector, use **matplotlib** subplot to create subplots to fit all the sector stocks in one plot. One row can have a maximum of 3 charts.
 - c) Using your sector DataFrame use the `corr()` DataFrame method to come up with a correlogram. Create a DataFrame for these correlations
 - d) Use **Seaborn** to plot the **correlation plot** for your sector stocks.

Key performance Metrics:

- Go an extra step to produce charts that are visually appealing
- Ensure all the plots have a Title
- Ensure all plots have x labels and y labels where applicable
- Your plots should be clearly visible. Change the size of your plot to a comfortable width and height.
- Save all your plots

```
In [8]: import os
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

Ensure that you have the ***clean_stock_prices.csv*** file in your working directory

```
In [12]: os.listdir()
```

```

Out[12]: ['.ipynb_checkpoints',
'ACFrOgDNOpXraTWAiHYnIJsSZj9PkjkTe9jHRVpW_lhLtlGyJe-iUVQT_0bi4gXfb2XUhorNaclA_1cu
yp9TaPguf0_VF01k4Dp_dcrT6SV0-3MpU8Ry7bQjP1YKOpM=.pdf',
'Anaconda3-2021.05-Windows-x86_64.exe',
'arduino-ide_2.0.0-rc4_Windows_64bit.exe',
'AutoCAD_2022_English_Win_64bit_di_en-US_setup_webinstall.exe',
'AutoCAD_2022_English_Win_64bit_dlm.sfx.exe',
'banking_stock_prices.png',
'BRILLIANT KCPE SCIENCE STD 8.pdf',
'ChromeSetup.exe',
'Class-8-Science-End-Term-1-2020..pdf',
'clean_stock_data.csv',
'clean_stock_prices.csv',
'Click_me_to_install_SnapTube_tube_organic_search.apk',
'Click_me_to_install_SnapTube_tube_snaptube.com.apk',
'codeblocks-20.03mingw-setup.exe',
'Course Syllabus.docx',
'desktop.ini',
'MacGyver 2016 Season 1 Complete HDTV x264 [i_c]',
'Major Lazer - Light it Up Remix (feat. Nyla & Fuse ODG) (Music Video) by Method
Studios.webm',
'MathcadPrime1.0',
'MathcadPrime1.0.zip',
'MATHS-4.pdf',
'matlab_R2021b_win64.exe',
'musicmatters_yearmix_2019_2020_tracklist_h264_35953.mp4',
'mysql-installer-web-community-8.0.28.0.msi',
'new_daily_prices.csv',
'Politics and Policy Final Project Dataset (1).xlsx',
'Politics and Policy Final Project Dataset.xlsx',
'project-time-series-workbook (1).ipynb',
'project-time-series-workbook (2).ipynb',
'project-time-series-workbook.ipynb',
'PTC Mathcad Prime 6.0.0.0 [FileCR].zip',
'python-3.10.2-amd64.exe',
'R-4.1.2-win.exe',
'rarreg.key',
'RStudio-2021.09.2-382.exe',
'Snaptube_v5.18.0.5183310_apkpure.com.apk',
'SOCIAL-3.pdf',
'SpotifySetup.exe',
'Star Wars All Movies Collection (1977-2015) 720p Bluray x264 Dual Audio [Hindi-E
nglish] - KartiKing',
'Statistical tables-20211109.zip',
'Student workbook',
'student_copy_pandas_workbook.ipynb',
'student_workbook_stocks.ipynb',
'Telegram Desktop',
'top-10-brands.png',
'tsetup.2.8.4.exe',
'uTorrent.exe',
'vehicle_data (1).csv',
'vehicle_data (2).csv',
'vehicle_data (3).csv',
'vehicle_data.csv',
'vehicle_dataset_project (1).ipynb',
'vehicle_dataset_project (2).ipynb',
'vehicle_dataset_project.ipynb',
'videoplayback (1).mp4',
'videoplayback.mp4',
'VSPER theory and molecular shapes.edited (1) 2.docx',
'WhatsApp Image 2021-07-21 at 3.43.38 PM.jpeg',
'WhatsApp Image 2021-07-21 at 3.43.39 PM.jpeg',
'WhatsApp Image 2022-02-05 at 20.35.30.jpeg',

```

```
'WhatsAppSetup (1).exe',
'WhatsAppSetup.exe',
'winrar-x64-602.exe',
'[1337x.buzz]__Lucifer.S05.480p.x264-ZMNT.torrent',
'_Getintopc.com_PTC.Mathcad.Prime.6.0.0.0.Win64',
'_Getintopc.com_PTC.Mathcad.Prime.6.0.0.0.Win64.rar',
'_temp_matlab_R2021b_win64']
```

If you can see the `clean_stock_prices.csv` as an output in the above cell, read the data into a DataFrame using pandas

```
In [13]: # read in the necessary file ('clean_stock_prices.csv')
df = pd.read_csv('clean_stock_prices.csv', index_col=0)
df.head()
```

```
Out[13]:
```

	EGAD	KUKZ	LIMIT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	
Date														
2022-01-13	12.90	385.0	320.0	22.20	130.00	54.00	11.80	30.00	59.00	49.55	...	440.0	10.80	1
2022-01-11	13.80	385.0	320.0	20.55	134.75	44.75	11.90	30.75	59.50	52.00	...	445.0	10.85	1
2022-01-07	13.80	420.0	320.0	21.25	132.00	37.05	11.80	29.05	60.00	53.00	...	442.0	10.90	1
2022-01-06	13.80	420.0	320.0	20.25	130.75	33.70	11.80	29.30	60.00	53.00	...	442.0	10.90	1
2022-01-05	12.85	420.0	320.0	19.95	130.75	30.60	11.75	29.50	59.75	53.00	...	442.0	10.90	1

5 rows × 60 columns

```
In [14]: df.tail()
```

```
Out[14]:
```

	EGAD	KUKZ	LIMIT	SASN	WTK	CGEN	ABSA	BKG	DTK	EQTY	...	BAT	CARB	
Date														
2021-08-09	12.15	415.0	300.00	19.50	134.5	35.0	9.80	32.40	65.75	50.25	...	445.5	12.25	1
2021-08-06	12.15	415.0	300.00	20.00	134.5	35.0	9.80	32.40	65.75	50.00	...	454.0	12.25	1
2021-08-05	12.30	415.0	320.00	20.00	134.5	35.0	9.82	31.85	65.00	49.40	...	450.0	12.20	1
2021-08-04	12.00	415.0	320.00	19.95	135.0	35.0	9.76	29.75	64.00	49.10	...	455.0	12.00	1
2021-08-03	11.80	415.0	304.75	19.95	134.5	35.0	9.82	29.50	65.00	49.00	...	450.0	12.00	1

5 rows × 60 columns

Use this part to answer questions 1, 2 and 3

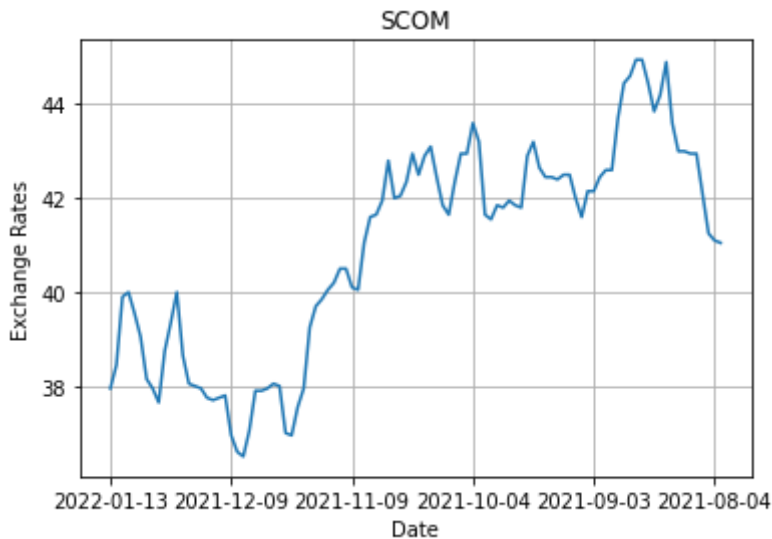
```
In [15]: # Lowest price for Safaricom
df['SCOM'].nsmallest(1)
```

```
Out[15]: Date
2021-12-07    36.5
Name: SCOM, dtype: float64
```

```
In [16]: # highest price for Safaricom
df['SCOM'].nlargest(1)
```

```
Out[16]: Date
2021-08-24    44.95
Name: SCOM, dtype: float64
```

```
In [19]: # Plot SCOM to confirm above observations
df['SCOM'].plot()
plt.grid()
plt.xlabel('Date')
plt.ylabel('Exchange Rates')
plt.title('SCOM')
plt.show()
```



Use this part to answer question 4

```
In [ ]: # agricultural companies
agric = ['EGAD', 'KUKZ', 'LIMT', 'SASN', 'WTK']

# commercial companies
comm = ['XPRS', 'KQ', 'LKL', 'NBV', 'NMG', 'SMER', 'SCAN', 'SGL', 'TPSE', 'UCHM']

# banking companies
bank = ['ABSA', 'BKG', 'DTK', 'EQTY', 'HFCK', 'IMH', 'KCB', 'NBK', 'NCBA', 'SBIC', 'SCBK',

# construction sector
const = ['ARM', 'BAMB', 'CRWN', 'CABL', 'PORT']

# energy sector
energy = ['KEGN', 'KPLC', 'TOTL', 'UMME']

# insurance sector
insur = ['BRIT', 'CIC', 'JUB', 'KNRE', 'LBTY', 'SLAM']

# investement sector
invest = ['CTUM', 'HAFR', 'KURV', 'OCH', 'TCL', 'NSE']
```

```
# manufacturing sector
manu = ['BOC', 'BAT', 'CARB', 'EABL', 'EVRD', 'FTGH', 'ORCH', 'MSC', 'UNGA']
```

To subset a sector simply use the **slice** notation. For example if I choose the Insurance sector, i will use the **insur** list

```
In [22]: insur_df = df.loc[:, 'BRIT': 'SLAM'].copy()
insur_df.head()
```

```
Out[22]:
```

	BRIT	CIC	JUB	KNRE	LBTY	SLAM
	Date					
2022-01-13	7.26	2.17	310.00	2.27	7.00	10.50
2022-01-11	7.14	2.17	310.00	2.32	7.00	10.60
2022-01-07	7.52	2.13	310.00	2.30	7.04	11.55
2022-01-06	7.52	2.15	310.50	2.29	7.04	11.55
2022-01-05	7.50	2.10	316.75	2.30	7.04	11.55

```
In [23]: insur_cols = insur_df.columns
for insurance in insur_cols:
    print(insurance)
```

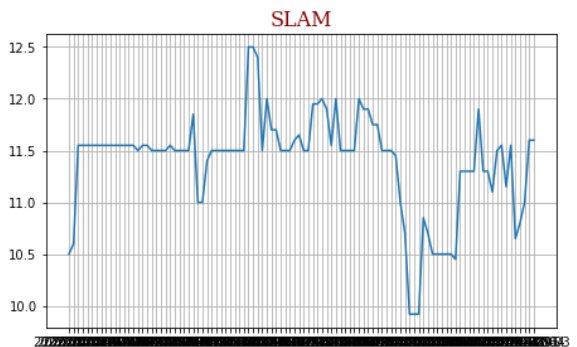
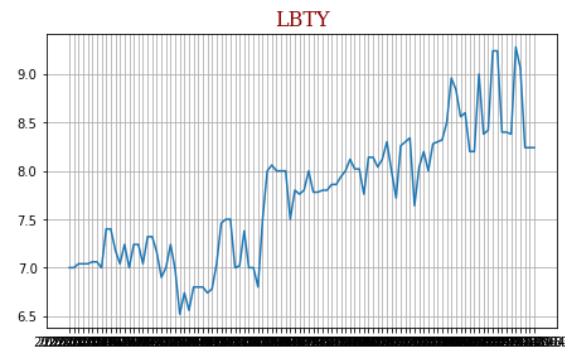
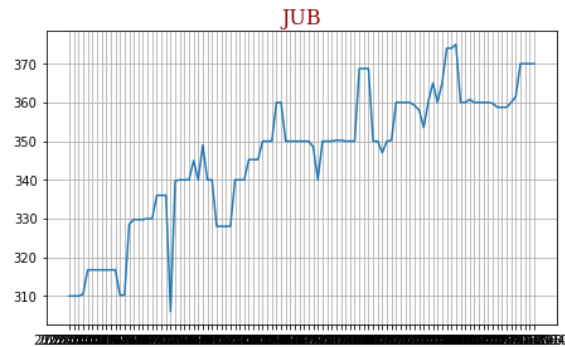
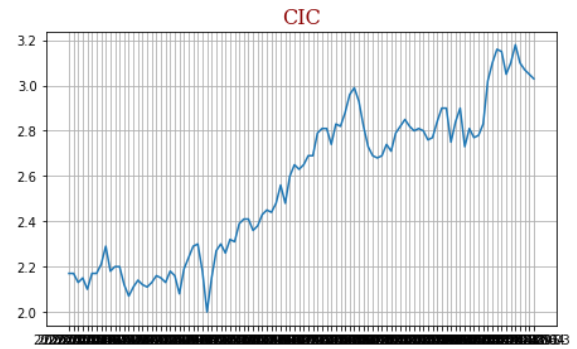
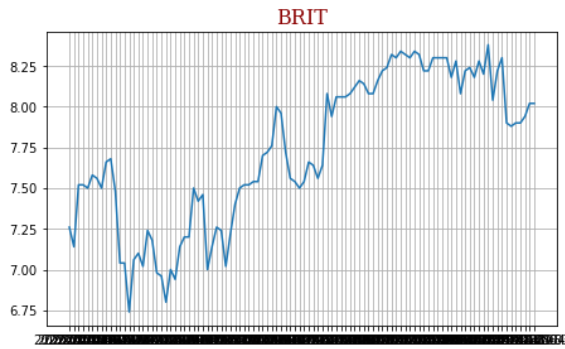
```
BRIT
CIC
JUB
KNRE
LBTY
SLAM
```

```
In [25]: insur_cols = insur_df.columns

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx, insurance in enumerate(insur_cols, start=1):
    plt.subplot(6, 2, idx)
    plt.title(insurance, fontdict=font)
    plt.grid()
    plt.plot(insurance, data=df)

fig = plt.gcf()
fig.set_size_inches(16, 30)
plt.show()
```



```
In [26]: agric_df=df.loc[:, 'EGAD': 'WTK'].copy()
         agric_df.head()
```

```
Out[26]:
```

	EGAD	KUKZ	LIMT	SASN	WTK
Date					
2022-01-13	12.90	385.0	320.0	22.20	130.00
2022-01-11	13.80	385.0	320.0	20.55	134.75
2022-01-07	13.80	420.0	320.0	21.25	132.00
2022-01-06	13.80	420.0	320.0	20.25	130.75
2022-01-05	12.85	420.0	320.0	19.95	130.75

```
In [27]: agric_cols=agric_df.columns
         for agriculture in agric_cols:
           print(agriculture)
```

```
EGAD
KUKZ
LIMT
SASN
WTK
```

```
In [28]: agric_cols = agric_df.columns

         font = {'family': 'serif',
                 'color': 'darkred',
```

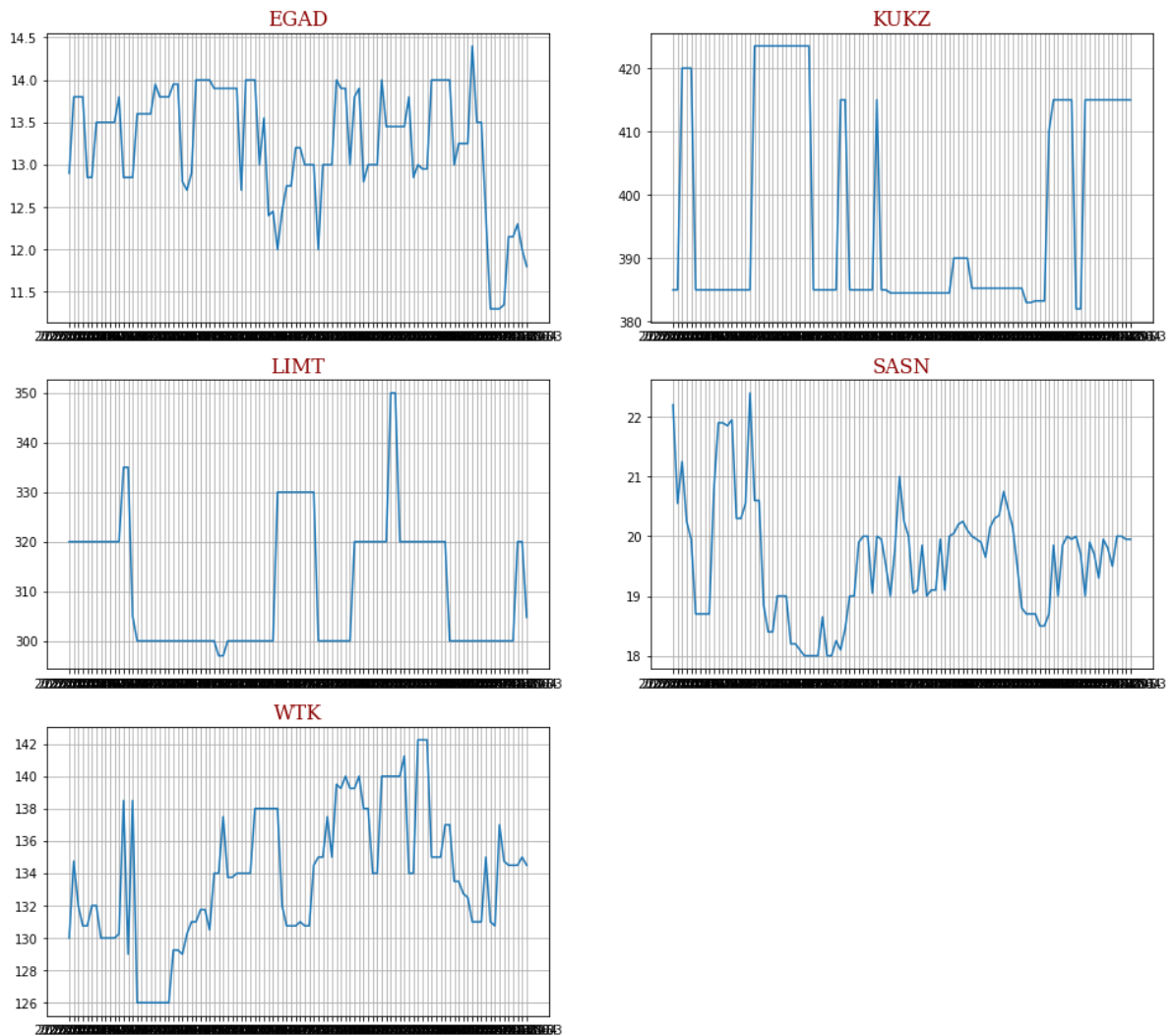
```

    'weight': 'normal',
    'size': 16,
}

for idx,agriculture in enumerate(agric_cols,start=1):
    plt.subplot(6,2,idx)
    plt.title(agriculture,fontdict=font)
    plt.grid()
    plt.plot(agriculture,data=df)

fig = plt.gcf()
fig.set_size_inches(16,30)
plt.show()

```



```

In [29]: comm_df=df.loc[:, 'XPRS': 'UCHM'].copy()
         comm_df.head()

```

```

Out[29]:

```

	XPRS	KQ	LKL	NBV	NMG	SMER	SCAN	SGL	TPSE	UCHM
Date										
2022-01-13	3.37	3.83	3.98	5.12	18.55	2.75	4.10	14.10	15.00	0.23
2022-01-11	3.50	3.83	4.00	5.30	19.00	2.82	4.14	15.50	15.00	0.21
2022-01-07	3.90	3.83	3.99	4.94	19.45	2.78	4.18	14.90	15.25	0.23
2022-01-06	4.10	3.83	4.00	4.89	19.90	2.80	4.12	14.90	15.25	0.23
2022-01-05	4.10	3.83	4.00	5.06	19.40	2.80	4.16	13.55	15.25	0.23


```
In [37]: comm_cols=comm_df.columns
for commercial in comm_cols:
    print(commercial)
```

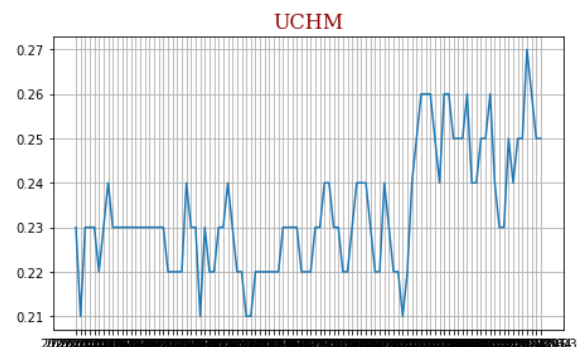
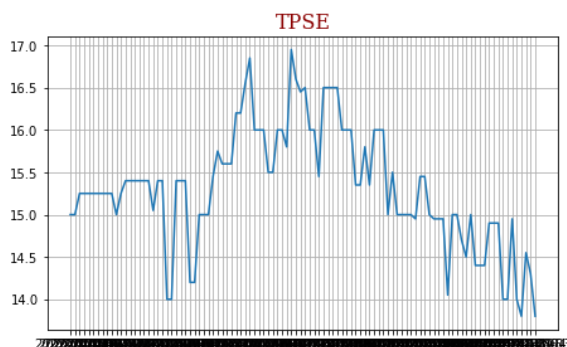
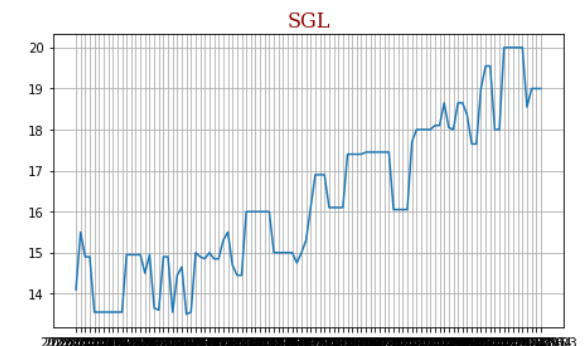
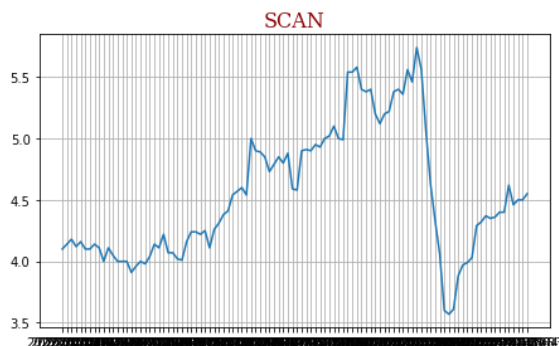
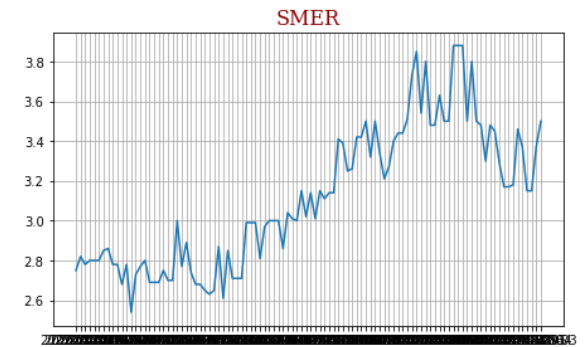
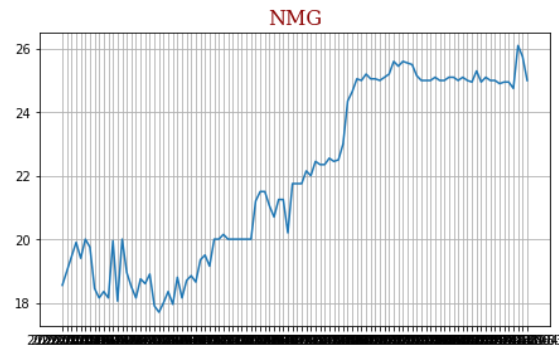
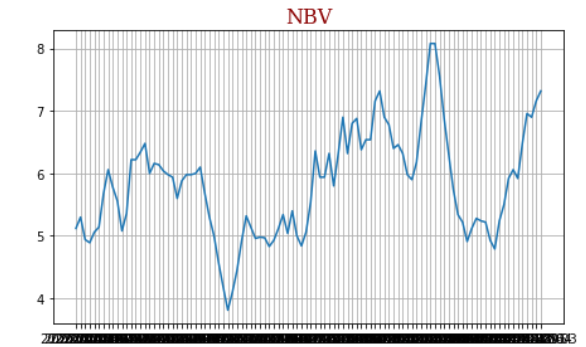
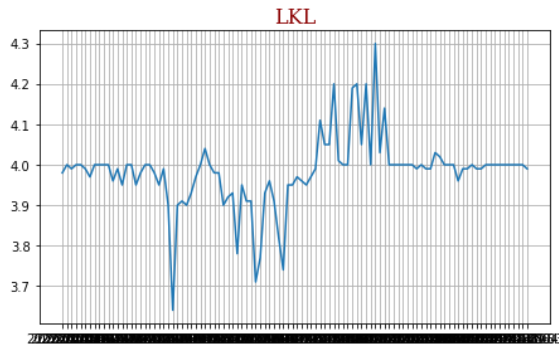
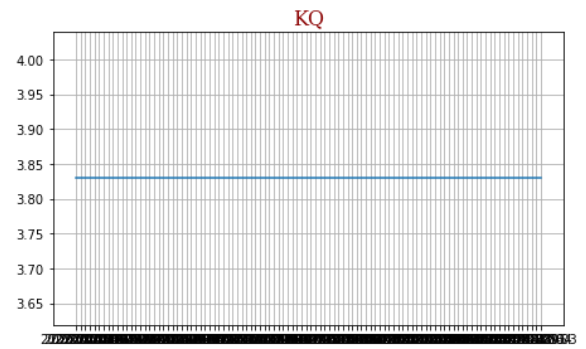
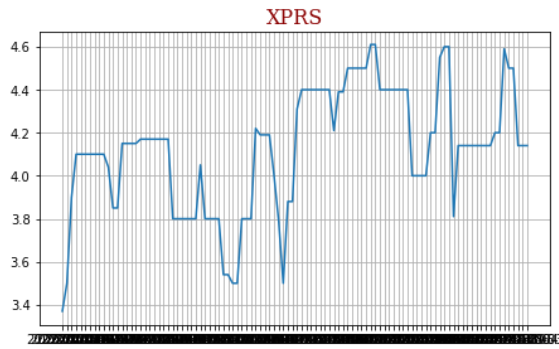
XPRS
KQ
LKL
NBV
NMG
SMER
SCAN
SGL
TPSE
UCHM

```
In [38]: comm_cols = comm_df.columns

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx, commercial in enumerate(comm_cols, start=1):
    plt.subplot(6,2,idx)
    plt.title(commercial, fontdict=font)
    plt.grid()
    plt.plot(commercial, data=df)

fig = plt.gcf()
fig.set_size_inches(16,30)
plt.show()
```



```
In [33]: bank_df=df.loc[:, 'ABSA': 'COOP'].copy()
bank_df.head()
```

Out[33]:

	ABSA	BKG	DTK	EQTY	HFCK	IMH	KCB	NBK	NCBA	SBIC	SCBK	COOP
Date												
2022-01-13	11.80	30.00	59.00	49.55	3.64	21.00	45.25	4.12	25.70	88.5	129.50	12.55
2022-01-11	11.90	30.75	59.50	52.00	3.81	21.50	45.85	4.12	25.95	87.5	130.00	12.80
2022-01-07	11.80	29.05	60.00	53.00	3.81	21.40	46.00	4.12	25.95	87.0	130.50	12.95
2022-01-06	11.80	29.30	60.00	53.00	3.89	21.45	45.90	4.12	25.90	87.0	130.75	13.00
2022-01-05	11.75	29.50	59.75	53.00	3.81	21.45	45.50	4.12	25.55	87.0	130.00	13.00

In [35]:

```
bank_cols=bank_df.columns
for banking in bank_cols:
    print(banking)
```

ABSA
BKG
DTK
EQTY
HFCK
IMH
KCB
NBK
NCBA
SBIC
SCBK
COOP

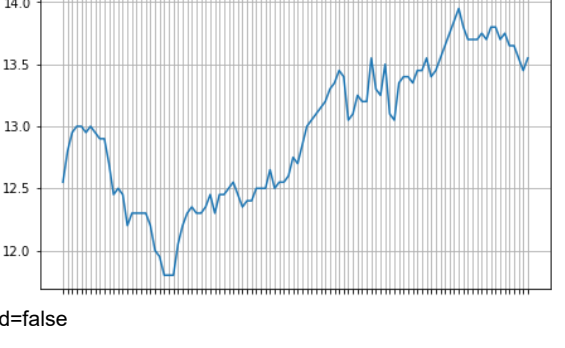
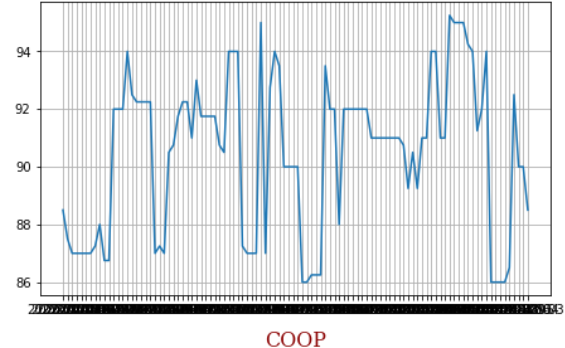
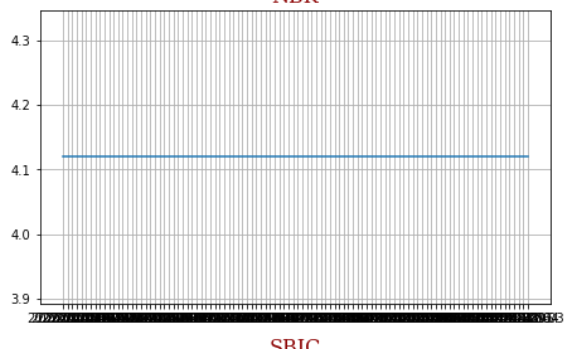
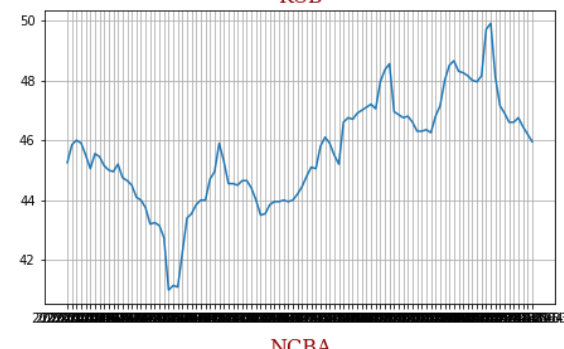
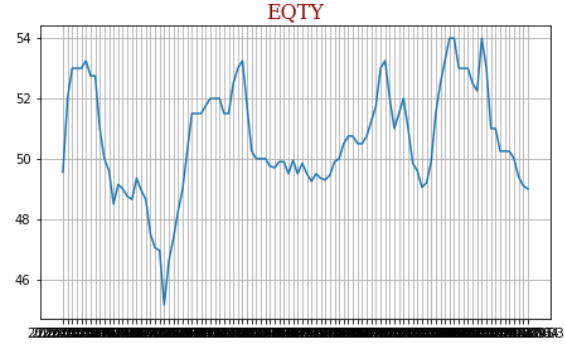
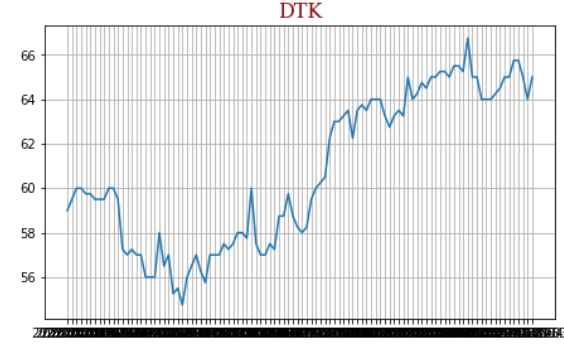
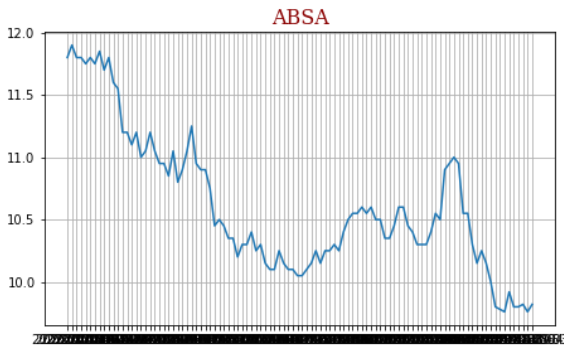
In [36]:

```
bank_cols = bank_df.columns

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx, banking in enumerate(bank_cols, start=1):
    plt.subplot(6,2,idx)
    plt.title(banking, fontdict=font)
    plt.grid()
    plt.plot(banking, data=df)

fig = plt.gcf()
fig.set_size_inches(16,30)
plt.show()
```



```
In [39]: const_df=df.loc[:, 'ARM':'PORT'].copy()
const_df.head()
```

```
Out[39]:
```

	ARM	BAMB	CRWN	CABL	PORT
Date					
2022-01-13	5.55	38.05	30.75	1.19	7.02
2022-01-11	5.55	37.75	30.50	1.20	7.20
2022-01-07	5.55	38.00	30.50	1.21	7.00
2022-01-06	5.55	37.85	30.75	1.27	7.02
2022-01-05	5.55	37.95	30.50	1.22	6.90

```
In [40]: const_cols=const_df.columns
for construction in const_cols:
    print(construction)
```

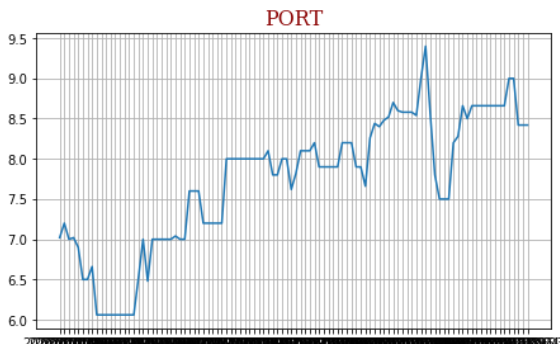
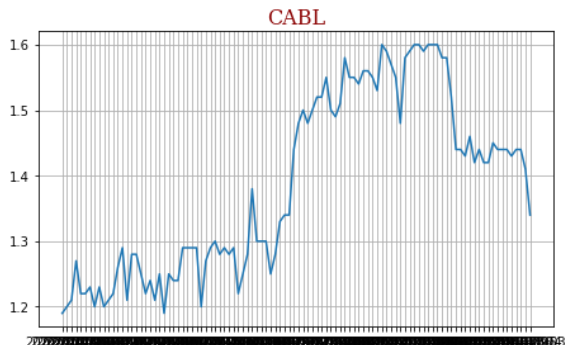
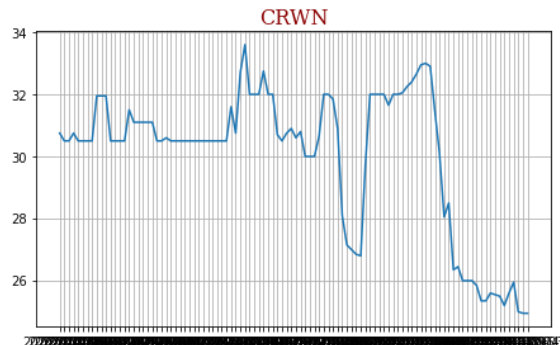
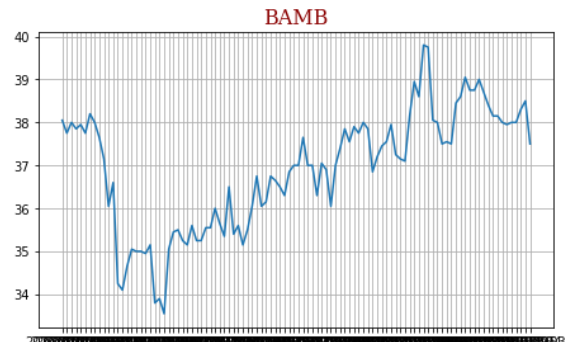
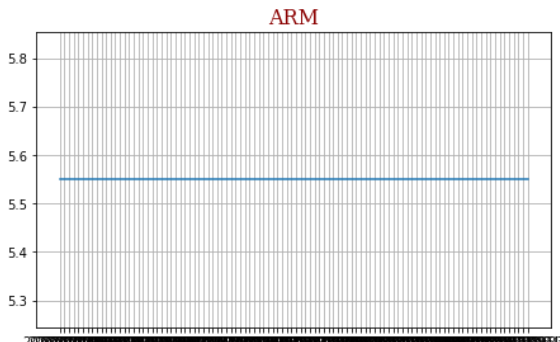
ARM
BAMB
CRWN
CABL
PORT

```
In [41]: const_cols = const_df.columns

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx, construction in enumerate(const_cols, start=1):
    plt.subplot(6,2,idx)
    plt.title(construction, fontdict=font)
    plt.grid()
    plt.plot(construction, data=df)

fig = plt.gcf()
fig.set_size_inches(16,30)
plt.show()
```



```
In [42]: energ_df=df.loc[:, 'KEGN': 'UMME']
energ_df.head()
```

```
Out[42]:
```

	KEGN	KPLC	TOTL	UMME
Date				
2022-01-13	4.15	1.67	24.80	6.72
2022-01-11	4.13	1.71	24.20	6.74
2022-01-07	4.12	1.72	24.60	6.74
2022-01-06	4.13	1.72	24.55	6.74
2022-01-05	4.16	1.71	24.70	6.74

```
In [43]: energ_cols=energ_df.columns
for energy in energ_cols:
    print(energy)
```

```
KEGN
KPLC
TOTL
UMME
```

```
In [44]: energ_cols = energ_df.columns

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
```

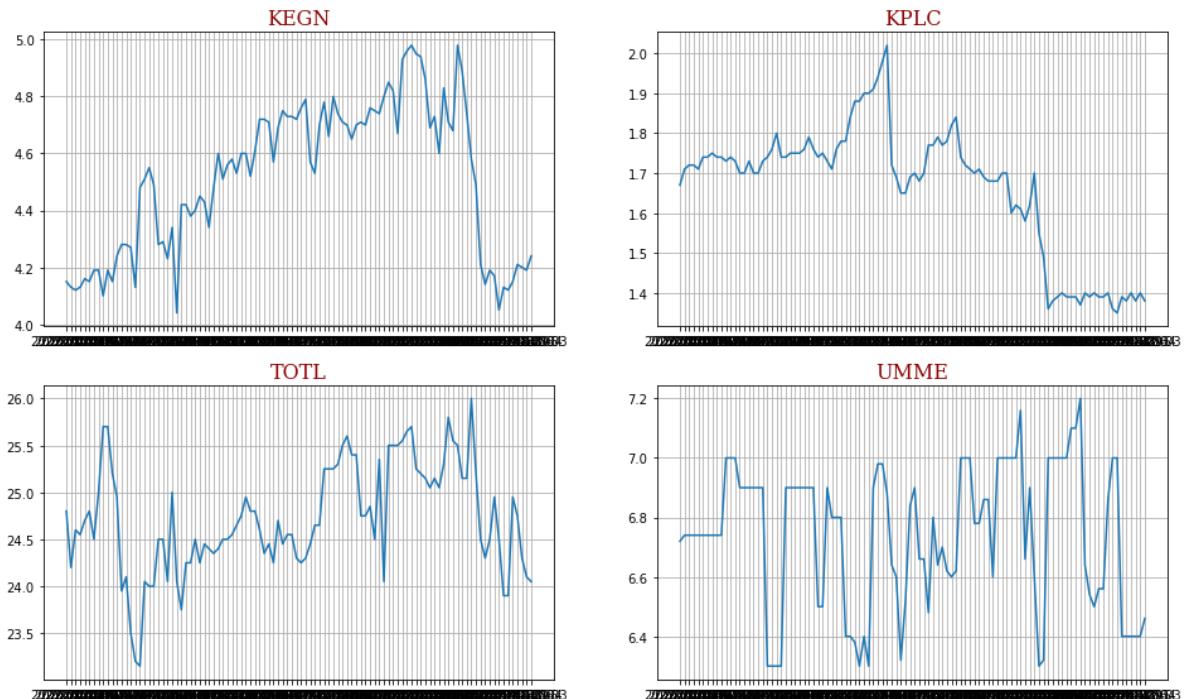
```

        'size': 16,
    }

    for idx,energy in enumerate(energ_cols,start=1):
        plt.subplot(6,2,idx)
        plt.title(energy,fontdict=font)
        plt.grid()
        plt.plot(energy,data=df)

    fig = plt.gcf()
    fig.set_size_inches(16,30)
    plt.show()

```



```

In [45]: invest_df=df.loc[:, 'CTUM': 'NSE'].copy()
         invest_df.head()

```

```

Out[45]:

```

	CTUM	HAFR	KURV	OCH	TCL	NSE
Date						
2022-01-13	14.65	0.38	1500.0	1.80	1.36	8.36
2022-01-11	14.35	0.40	1500.0	1.84	1.36	8.26
2022-01-07	14.40	0.40	1500.0	1.88	1.36	8.16
2022-01-06	14.50	0.38	1500.0	1.97	1.32	8.20
2022-01-05	14.60	0.39	1500.0	1.97	1.29	8.12

```

In [46]: invest_cols=invest_df.columns
         for investment in invest_cols:
             print(investment)

```

```

CTUM
HAFR
KURV
OCH
TCL
NSE

```

```

In [47]: invest_cols = invest_df.columns

```

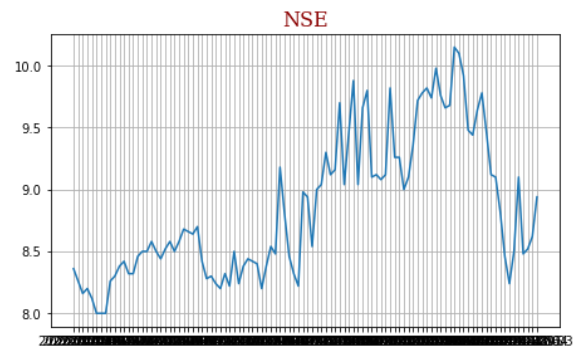
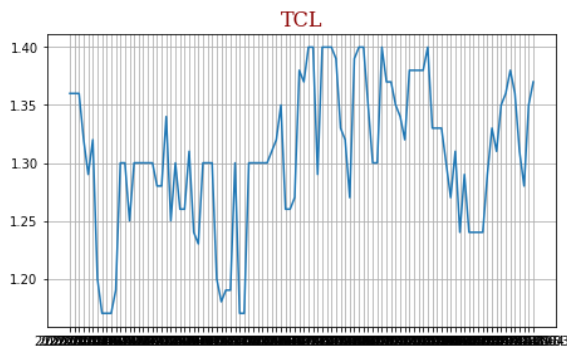
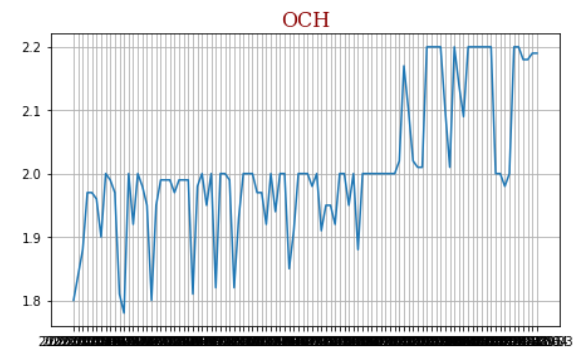
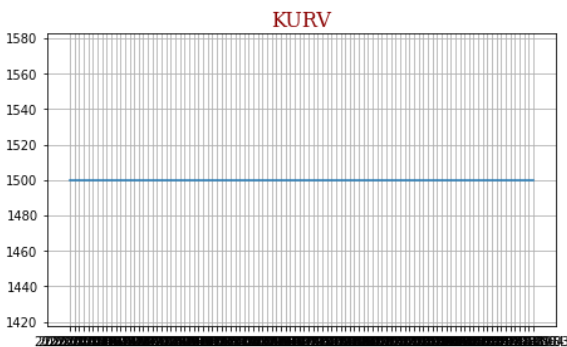
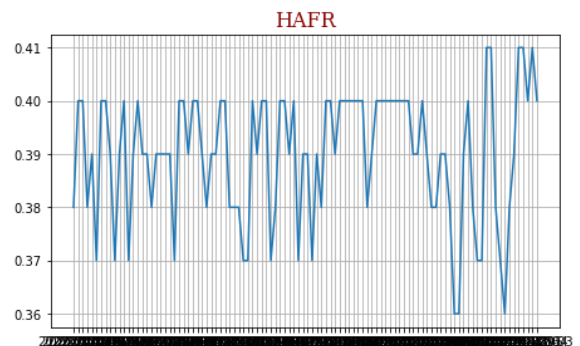
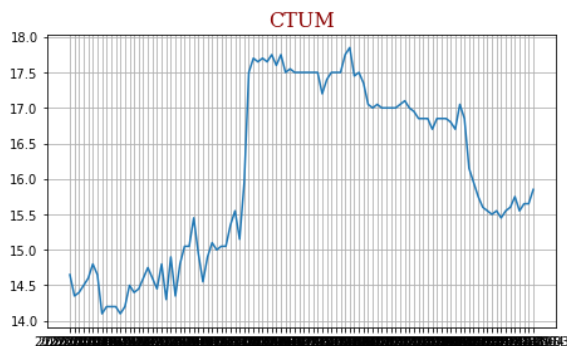
```

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx, investment in enumerate(invest_cols, start=1):
    plt.subplot(6,2,idx)
    plt.title(investment, fontdict=font)
    plt.grid()
    plt.plot(investment, data=df)

fig = plt.gcf()
fig.set_size_inches(16,30)
plt.show()

```



```

In [48]: manu_df=df.loc[:, 'BOC':'UNGA'].copy()
manu_df.head()

```


Out[48]:

	BOC	BAT	CARB	EABL	EVRD	FTGH	ORCH	MSC	UNGA
Date									
2022-01-13	72.5	440.0	10.80	151.50	0.96	1.34	10.4	0.27	27.10
2022-01-11	73.0	445.0	10.85	161.00	0.88	1.31	10.4	0.27	27.65
2022-01-07	73.0	442.0	10.90	164.75	0.94	1.30	10.4	0.27	27.65
2022-01-06	72.0	442.0	10.90	160.75	0.99	1.29	10.4	0.27	27.65
2022-01-05	70.0	442.0	10.90	163.75	0.99	1.26	10.4	0.27	27.65

```
In [49]: manu_cols=manu_df.columns
for manufacturing in manu_cols:
    print(manufacturing)
```

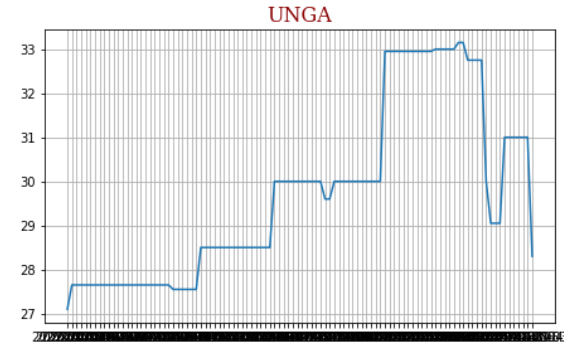
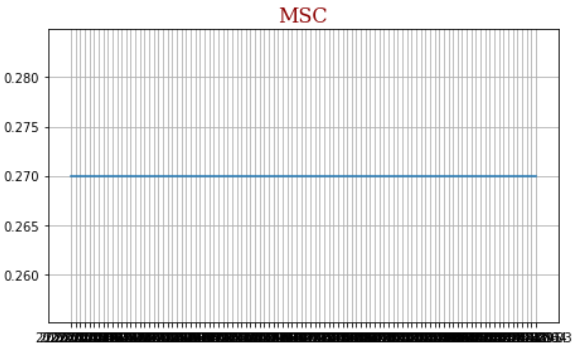
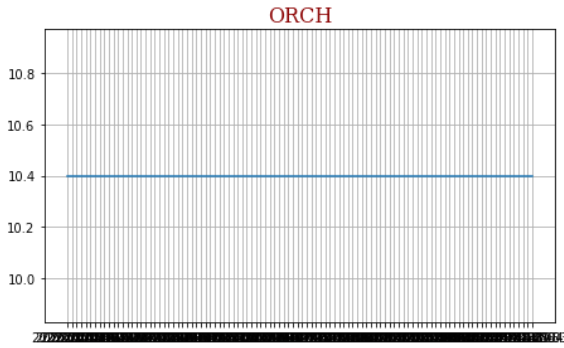
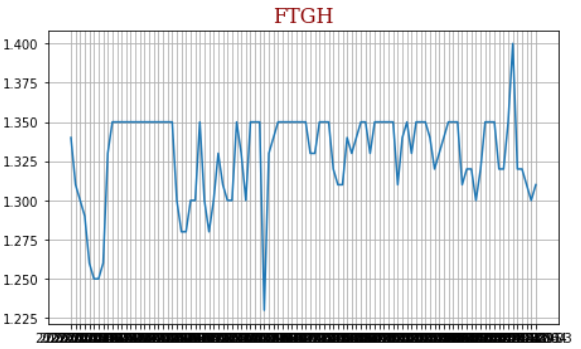
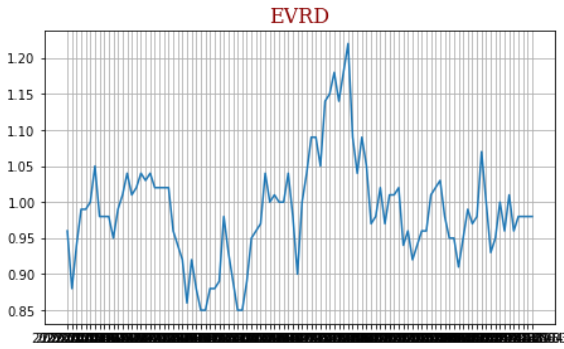
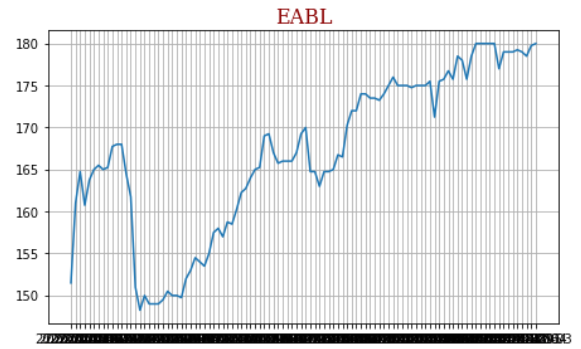
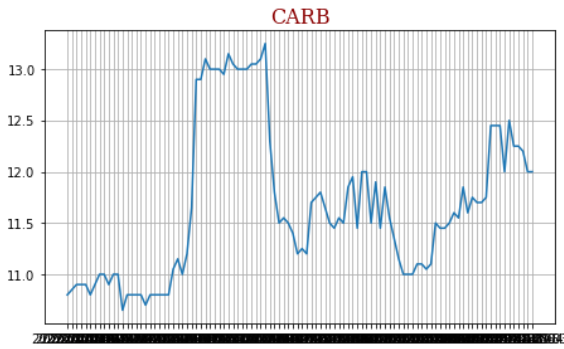
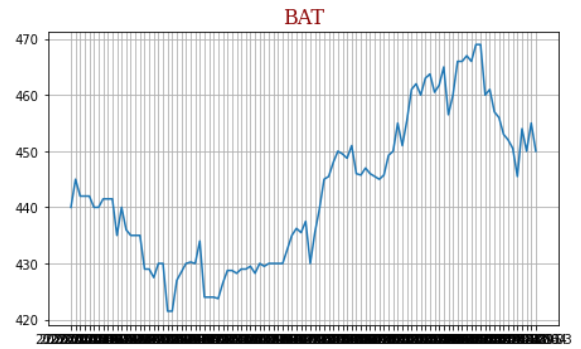
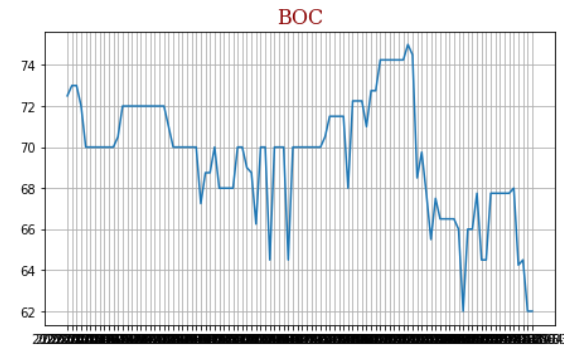
BOC
BAT
CARB
EABL
EVRD
FTGH
ORCH
MSC
UNGA

```
In [50]: manu_cols = manu_df.columns

font = {'family': 'serif',
        'color': 'darkred',
        'weight': 'normal',
        'size': 16,
        }

for idx, manufacturing in enumerate(manu_cols, start=1):
    plt.subplot(6, 2, idx)
    plt.title(manufacturing, fontdict=font)
    plt.grid()
    plt.plot(manufacturing, data=df)

fig = plt.gcf()
fig.set_size_inches(16, 30)
plt.show()
```



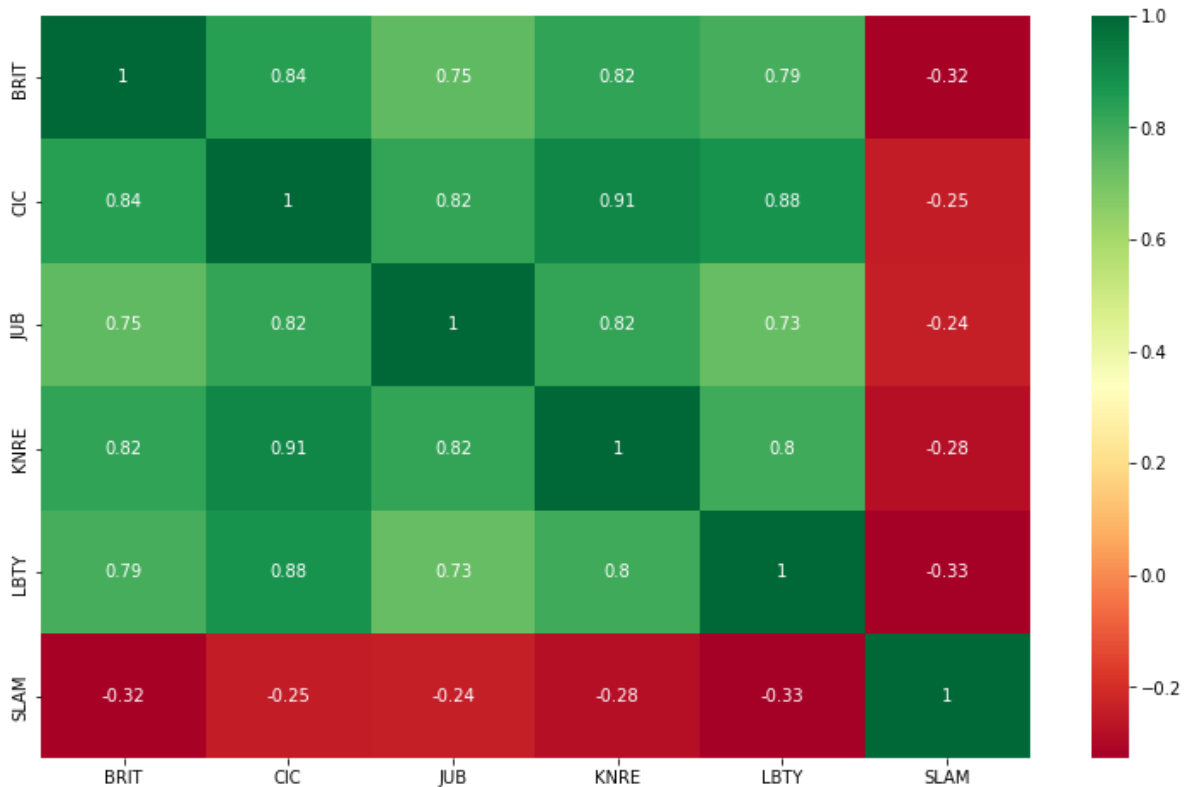
```
In [55]: insur_df.corr(method='pearson')
```

Out[55]:

	BRIT	CIC	JUB	KNRE	LBTY	SLAM
BRIT	1.000000	0.843570	0.745239	0.824231	0.791978	-0.318103
CIC	0.843570	1.000000	0.819647	0.914090	0.878433	-0.247342
JUB	0.745239	0.819647	1.000000	0.819330	0.734525	-0.241677
KNRE	0.824231	0.914090	0.819330	1.000000	0.800351	-0.282419
LBTY	0.791978	0.878433	0.734525	0.800351	1.000000	-0.326399
SLAM	-0.318103	-0.247342	-0.241677	-0.282419	-0.326399	1.000000

In [58]: `corr_df=insur_df.corr(method='pearson')`In [59]: `plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()`

Out[59]: <Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

In [60]: `agric_df.corr(method='pearson')`

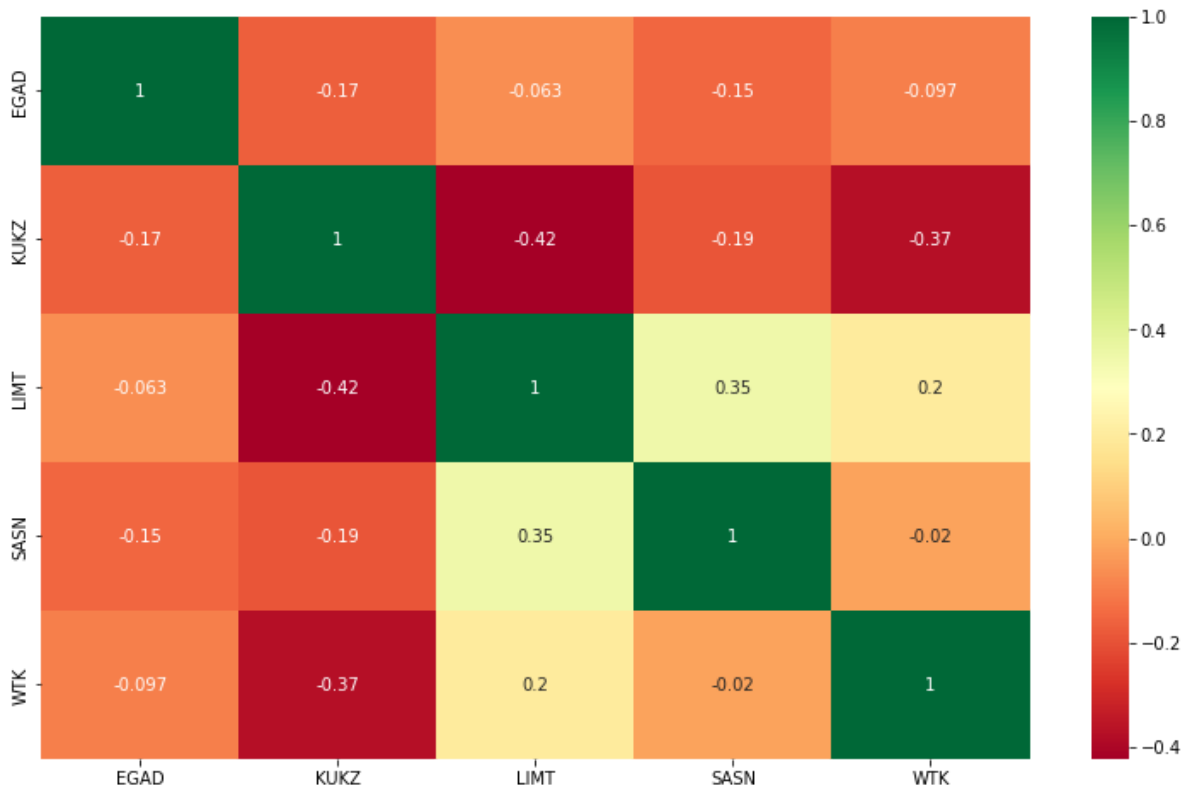
Out[60]:

	EGAD	KUKZ	LIMIT	SASN	WTK
EGAD	1.000000	-0.168294	-0.063075	-0.152460	-0.097436
KUKZ	-0.168294	1.000000	-0.422992	-0.192463	-0.374117
LIMIT	-0.063075	-0.422992	1.000000	0.345329	0.196757
SASN	-0.152460	-0.192463	0.345329	1.000000	-0.019554
WTK	-0.097436	-0.374117	0.196757	-0.019554	1.000000

In [61]: `corr_df=agric_df.corr(method='pearson')`

```
In [62]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

Out[62]: <Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

```
In [63]: comm_df.corr(method='pearson')
```

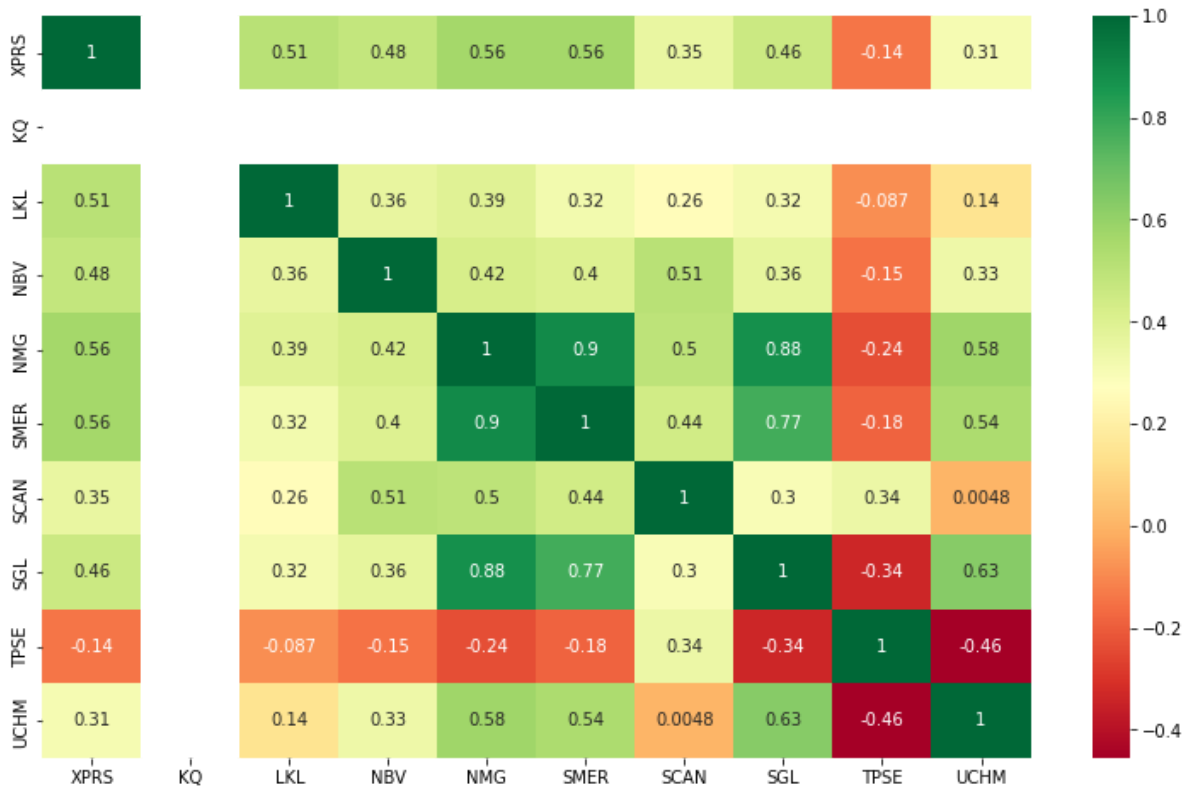
Out[63]:

	XPRS	KQ	LKL	NBV	NMG	SMER	SCAN	SGL	TPSE
XPRS	1.000000	NaN	0.505657	0.482448	0.564433	0.564130	0.353536	0.456735	-0.143478
KQ	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
LKL	0.505657	NaN	1.000000	0.361295	0.387344	0.318353	0.257131	0.317583	-0.087045
NBV	0.482448	NaN	0.361295	1.000000	0.423817	0.398074	0.508718	0.363434	-0.153862
NMG	0.564433	NaN	0.387344	0.423817	1.000000	0.896570	0.497322	0.879855	-0.235356
SMER	0.564130	NaN	0.318353	0.398074	0.896570	1.000000	0.441286	0.774443	-0.182699
SCAN	0.353536	NaN	0.257131	0.508718	0.497322	0.441286	1.000000	0.296204	0.344714
SGL	0.456735	NaN	0.317583	0.363434	0.879855	0.774443	0.296204	1.000000	-0.340694
TPSE	-0.143478	NaN	-0.087045	-0.153862	-0.235356	-0.182699	0.344714	-0.340694	1.000000
UCHM	0.310362	NaN	0.142561	0.334953	0.575009	0.542615	0.004806	0.633963	-0.455251

```
In [64]: corr_df=comm_df.corr(method='pearson')
```

```
In [65]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

Out[65]: <Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

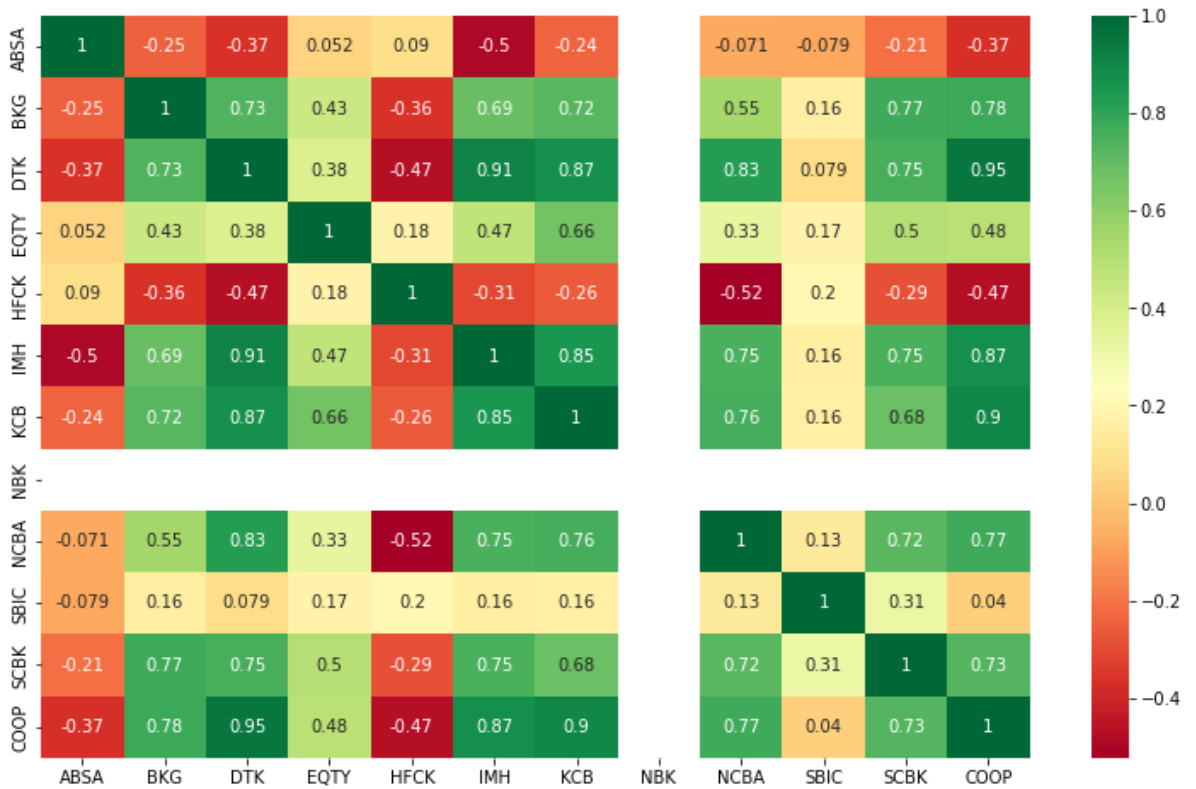
```
In [66]: bank_df.corr(method='pearson')
```

	ABSA	BKG	DTK	EQTY	HFCK	IMH	KCB	NBK	NCBA
ABSA	1.000000	-0.247357	-0.367356	0.051548	0.089564	-0.497121	-0.242094	NaN	-0.071353
BKG	-0.247357	1.000000	0.733606	0.431693	-0.363877	0.685030	0.722452	NaN	0.546149
DTK	-0.367356	0.733606	1.000000	0.377873	-0.472187	0.907300	0.865433	NaN	0.826353
EQTY	0.051548	0.431693	0.377873	1.000000	0.177967	0.468084	0.661149	NaN	0.333037
HFCK	0.089564	-0.363877	-0.472187	0.177967	1.000000	-0.312478	-0.263884	NaN	-0.522405
IMH	-0.497121	0.685030	0.907300	0.468084	-0.312478	1.000000	0.850515	NaN	0.748388
KCB	-0.242094	0.722452	0.865433	0.661149	-0.263884	0.850515	1.000000	NaN	0.761396
NBK	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NCBA	-0.071353	0.546149	0.826353	0.333037	-0.522405	0.748388	0.761396	NaN	1.000000
SBIC	-0.079066	0.160496	0.079410	0.173888	0.199610	0.159981	0.163069	NaN	0.134429
SCBK	-0.211586	0.769506	0.751985	0.495452	-0.288670	0.746789	0.679501	NaN	0.719180
COOP	-0.367982	0.777537	0.946788	0.484453	-0.469807	0.872273	0.902445	NaN	0.771551

```
In [67]: corr_df=bank_df.corr(method='pearson')
```

```
In [68]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

Out[68]: <Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

```
In [69]: const_df.corr(method='pearson')
```

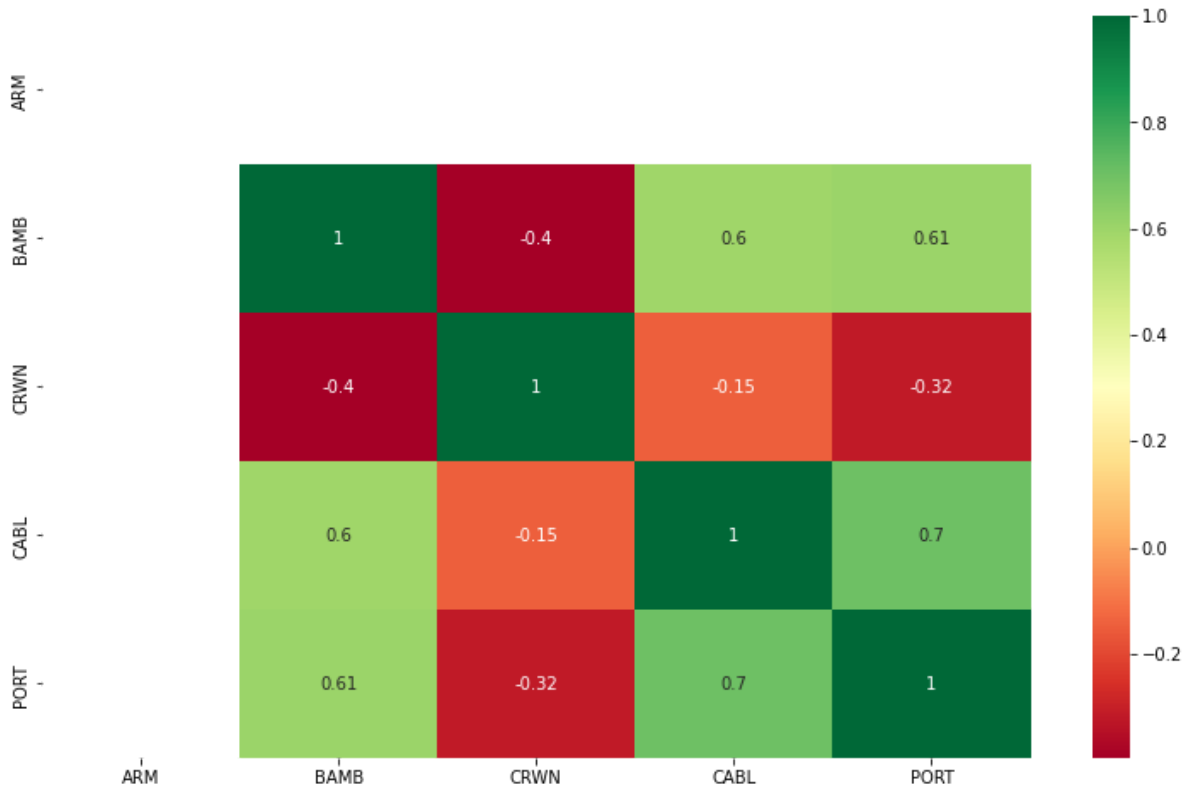
```
Out[69]:
```

	ARM	BAMB	CRWN	CABL	PORT
ARM	NaN	NaN	NaN	NaN	NaN
BAMB	NaN	1.000000	-0.395628	0.595477	0.607416
CRWN	NaN	-0.395628	1.000000	-0.147204	-0.315659
CABL	NaN	0.595477	-0.147204	1.000000	0.703981
PORT	NaN	0.607416	-0.315659	0.703981	1.000000

```
In [70]: corr_df=const_df.corr(method='pearson')
```

```
In [71]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

Out[71]: <Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

```
In [72]: energ_df.corr(method='pearson')
```

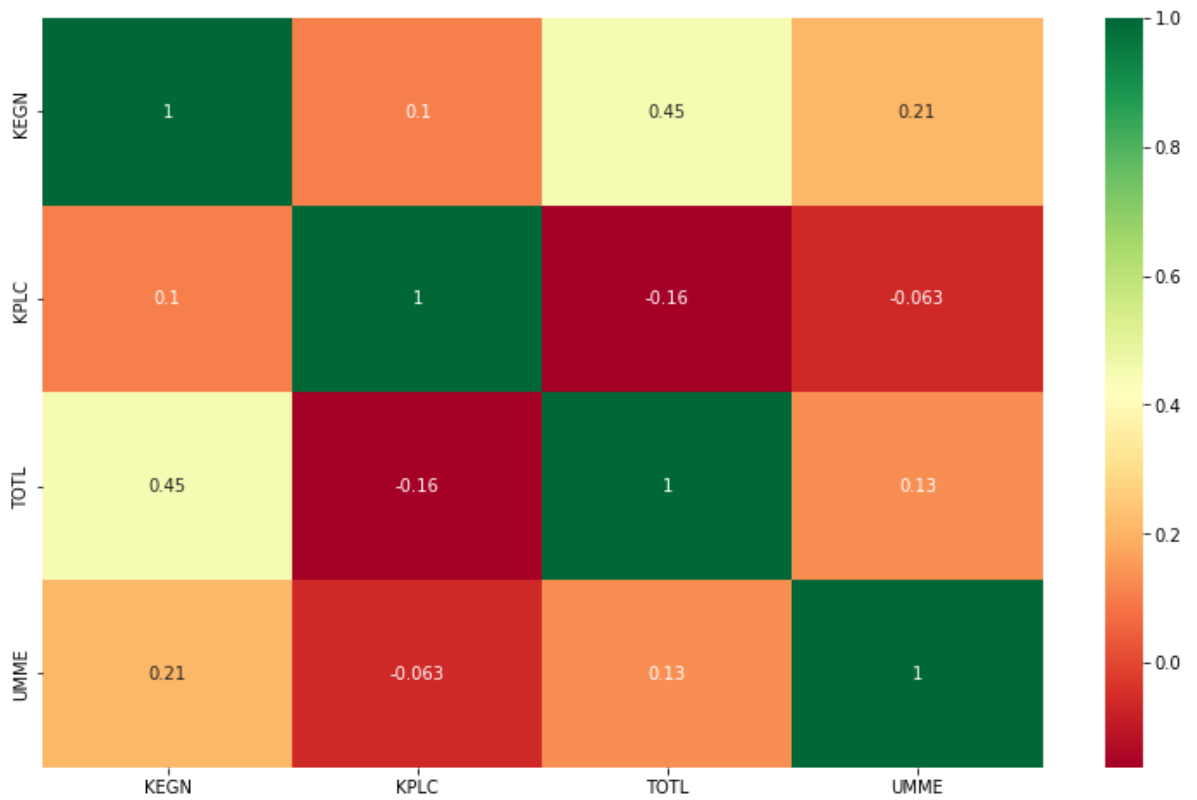
```
Out[72]:
```

	KEGN	KPLC	TOTL	UMME
KEGN	1.000000	0.104530	0.450142	0.206669
KPLC	0.104530	1.000000	-0.161972	-0.062976
TOTL	0.450142	-0.161972	1.000000	0.130735
UMME	0.206669	-0.062976	0.130735	1.000000

```
In [73]: corr_df=energ_df.corr(method='pearson')
```

```
In [74]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

```
Out[74]: <Figure size 432x288 with 0 Axes>
```



<Figure size 432x288 with 0 Axes>

```
In [75]: invest_df.corr(method='pearson')
```

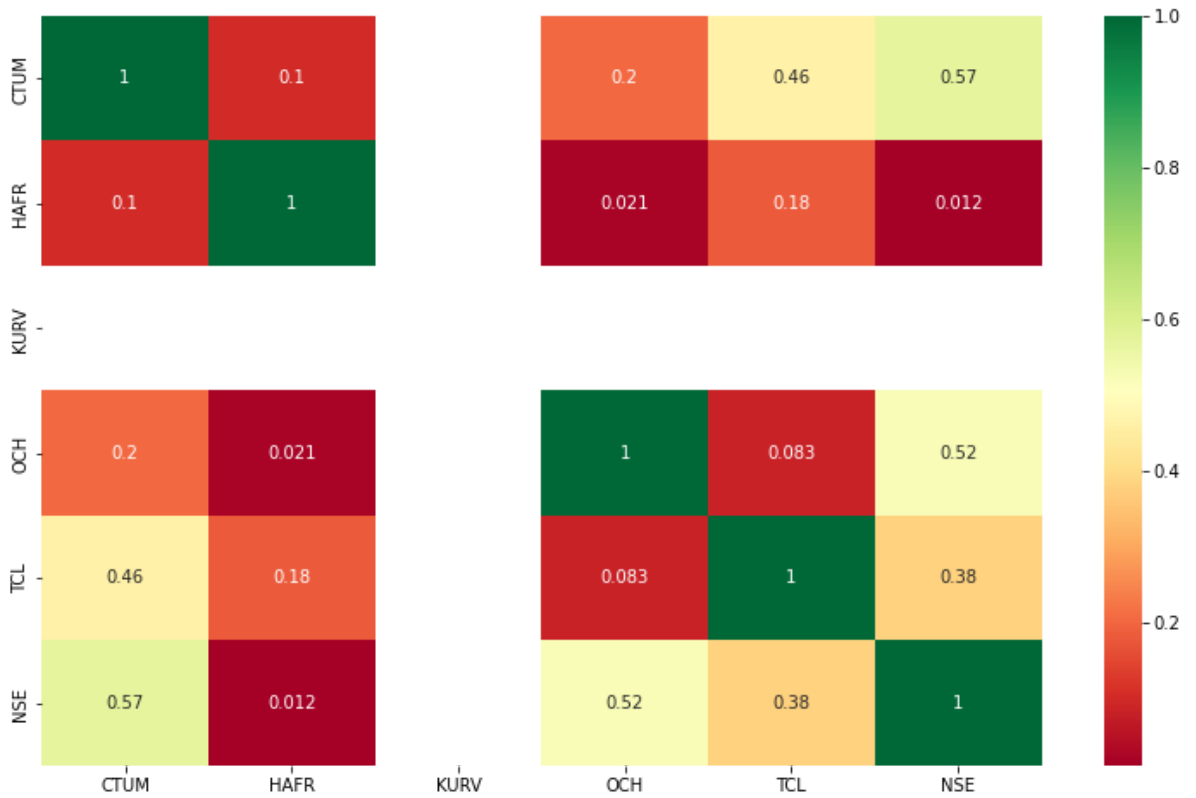
```
Out[75]:
```

	CTUM	HAFR	KURV	OCH	TCL	NSE
CTUM	1.000000	0.102660	NaN	0.201916	0.461223	0.568061
HAFR	0.102660	1.000000	NaN	0.021208	0.181057	0.011877
KURV	NaN	NaN	NaN	NaN	NaN	NaN
OCH	0.201916	0.021208	NaN	1.000000	0.083212	0.524421
TCL	0.461223	0.181057	NaN	0.083212	1.000000	0.381206
NSE	0.568061	0.011877	NaN	0.524421	0.381206	1.000000

```
In [76]: corr_df=invest_df.corr(method='pearson')
```

```
In [77]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

```
Out[77]: <Figure size 432x288 with 0 Axes>
```

<Figure size 432x288 with 0 Axes>

```
In [78]: manu_df.corr(method='pearson')
```

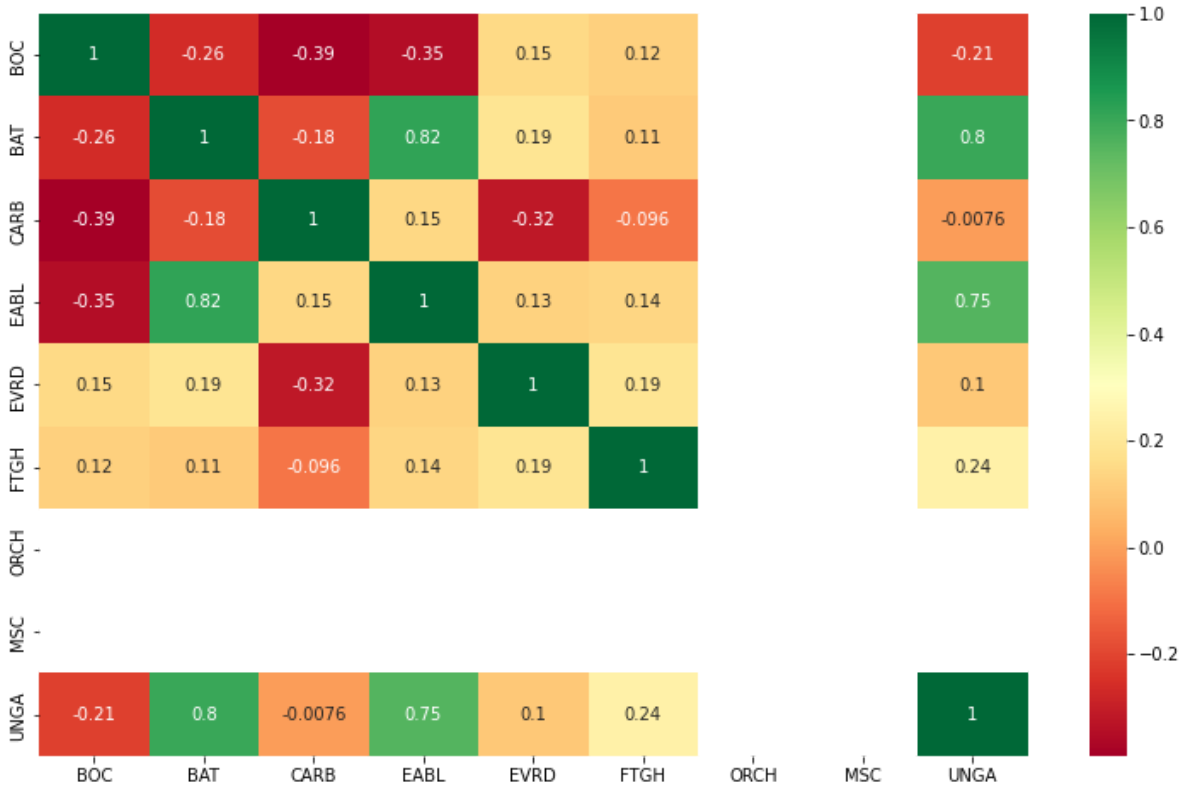
```
Out[78]:
```

	BOC	BAT	CARB	EABL	EVRD	FTGH	ORCH	MSC	UNGA
BOC	1.000000	-0.264702	-0.390657	-0.351283	0.147241	0.124104	NaN	NaN	-0.213949
BAT	-0.264702	1.000000	-0.184494	0.816630	0.187274	0.109096	NaN	NaN	0.802586
CARB	-0.390657	-0.184494	1.000000	0.146387	-0.317406	-0.095847	NaN	NaN	-0.007650
EABL	-0.351283	0.816630	0.146387	1.000000	0.128904	0.141361	NaN	NaN	0.753815
EVRD	0.147241	0.187274	-0.317406	0.128904	1.000000	0.187224	NaN	NaN	0.101246
FTGH	0.124104	0.109096	-0.095847	0.141361	0.187224	1.000000	NaN	NaN	0.243746
ORCH	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
MSC	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
UNGA	-0.213949	0.802586	-0.007650	0.753815	0.101246	0.243746	NaN	NaN	1.000000

```
In [79]: corr_df=manu_df.corr(method='pearson')
```

```
In [80]: plt.figure(figsize=(13, 8))
sns.heatmap(corr_df, annot=True, cmap='RdYlGn')
plt.figure()
```

```
Out[80]: <Figure size 432x288 with 0 Axes>
```



<Figure size 432x288 with 0 Axes>

In []:

Getting Started with Python

Vehicle Analysis Project

Author : fredrick odoyo

Date: 26/3/2022

Resources

1. ##### [Vehicle Dataset](#)
2. ##### [Submission Portal](#)

If you are having problems please refer to this document:

1. ##### [Data Analysis with Python Pandas Notebook](#)

Instructions

Import all the libraries listed in the first cell. Make sure all modules are installed.

Use the provided data set to answer the following:

Use `pandas` to come up with:

1. The titles and prices of **10** Cars with highest price
2. The titles and prices of 5 Buses & Microbuses with highest price
3. The titles and prices of 5 Trucks & Trailers with highest price

Plotting

Use `matplotlib` to come up with a plot indicating the **top 10 brands** that we have in the `vehicle_dataset`

Key performance Metrics:

- Ensure all the plots have a Title
- Ensure all plots have x labels and y labels where applicable
- Your plots should be clearly visible. Change the size of your plot to a comfortable width and height.

- Save all your plots

```
In [1]: import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
In [2]: os.listdir()
```

```
Out[2]: ['.ipynb_checkpoints',
'AMOS DOC..docx',
'AutoCAD_2022_English_Win_64bit_di_en-US_setup_webinstall.exe',
'Basic DC generator equations.pdf',
'CAT (1).pdf',
'CAT.pdf',
'clean_stock_prices.csv',
'codeblocks-20.03-setup.exe',
'DC Motors.pdf',
'desktop.ini',
'Document7 (2).pdf',
'Downloads - Shortcut.lnk',
'Efficiency and Power.pdf',
'Final_Project.pdf',
'Induction motors.pdf',
'introductory letter.pdf',
'Inventor_2022_ENUX64_DI_en-US_setup_webinstall.exe',
'Jack Donovan - The Way of Men (2012, Dissonant Hum) - libgen.lc.pdf',
'Lecture 3 Springs.pdf',
'lecture 4 notes.pdf',
'Lecture 4 Threaded fasteners.pdf',
'Lecture 5 - Welds part 1.pdf',
'Lecture 5.docx',
'Lecture 5.docx.pdf',
'Lecture 6 - Welds part 2.pdf',
'Lecture 6&7 (1).docx',
'Lecture 6&7.docx',
'Lecture 8 (1).docx',
'Lecture 8.docx',
'Lecture 9&10 (1).pdf',
'Lecture 9&10.pdf',
'matlab_R2021b_win64.exe',
'mysql-installer-web-community-8.0.28.0.msi',
'Power Amplifiers.pdf',
'project-time-series-workbook.ipynb',
'python-3.10.2-amd64.exe',
'python-for-data-science-sample-submission.pdf',
'Python_Week_One_Project (1).pdf',
'Python_Week_One_Project (2).pdf',
'Python_Week_One_Project.pdf',
'R Lesson 2.pptx',
'R-4.1.3-win.exe',
'R-and-R-Studio-Installation (1) (1).pdf',
'R-and-R-Studio-Installation (1) (2).pdf',
'R-and-R-Studio-Installation (1).pdf',
'Revit_2022_Ship_20210224_RTC_Win_64bit_di_ML_setup_webinstall.exe',
'Rollo Tomassi - The Rational Male-CreateSpace Independent Publishing Platform (2013).epub',
'Rollo Tomassi - The Rational Male-CreateSpace Independent Publishing Platform (2013).pdf',
'Series Amplifiers.pdf',
'student_copy_pandas_workbook.ipynb',
'TABLE 9.3.docx',
'TEAM CONTRACTS (1).docx',
'TEAM CONTRACTS.docx',
'vehicle_data.csv',
'vehicle_dataset_project.ipynb',
'[B07D23CFGR] James Clear - Atomic Habits (2018, Random House Business Books) - libgen.lc.pdf',
'[Microelectronics Journal vol. 29 iss. 8] V.B. Kitovski - Electronic devices and ci
```

rcuit theory, 6th edition, R. Boylestad and L. Nashelsky, Prentice Hall International Inc., 1996, 950 p (1998) [10.1016_s0026-2692(.pdf']

vehicle_data.csv should be listed in your output from the above cell

```
In [3]: df = pd.read_csv('vehicle_data.csv')
df.head()
```

```
Out[3]:
```

	title	category	region	parent_region	condition	attrs	brand	color	model	year
0	Toyota Land Cruiser Prado 2016 Black	Cars	Mvita	Mombasa	Foreign Used	First registration, No faults	Toyota	Black	Land Cruiser Prado	2016.
1	Mazda Demio 2014 Brown	Cars	Langata	Nairobi	Foreign Used	First owner, No faults	Mazda	Brown	Demio	2014.
2	Clean NV300 Caravan 2014 Model Dielsel 16 Seater	Buses & Microbuses	Kilimani	Nairobi	Foreign Used		Nissan	NaN	Caravan (Urvan)	2014.
3	Toyota Crown 2014 Pearl	Cars	Kilimani	Nairobi	Foreign Used	No faults	Toyota	Pearl	Crown	2014.
4	Honda Fit 2014 Black	Cars	Mvita	Mombasa	Foreign Used	No faults	Honda	Black	Fit	2014.

As an example I have shown the top 10 most expensive vehicles that are in parentregion Mombasa

```
In [5]: # filter only rows that have Cars as their category
df['category'] == 'Cars'
```

```
Out[5]:
```

0	True
1	True
2	False
3	True
4	True
...	
295	True
296	False
297	False
298	True
299	True

Name: category, Length: 300, dtype: bool

```
In [6]: mask = df['category'] == 'Cars'
```

```
In [7]: # all the rows in the DataFrame that have category cars
cars_df = df[mask].copy()
cars_df.head()
```

```
Out[7]:
```

	title	category	region	parent_region	condition	attrs	brand	color	model	y
0	Toyota Land Cruiser Prado 2016 Black	Cars	Mvita	Mombasa	Foreign Used	First registration, No faults	Toyota	Black	Land Cruiser Prado	20
1	Mazda Demio 2014 Brown	Cars	Langata	Nairobi	Foreign Used	First owner, No faults	Mazda	Brown	Demio	20
3	Toyota Crown 2014 Pearl	Cars	Kilimani	Nairobi	Foreign Used	No faults	Toyota	Pearl	Crown	20
4	Honda Fit 2014 Black	Cars	Mvita	Mombasa	Foreign Used	No faults	Honda	Black	Fit	20
5	Mitsubishi Delica 2013 White	Cars	Mvita	Mombasa	Foreign Used	First registration, No faults, Unpainted	Mitsubishi	White	Delica	20

To get the highest price I will use the `nlargest` function

```
In [8]: # top 10 cars with highest price
cars_df.nlargest(10, 'price')
```

Out[8]:

	title	category	region	parent_region	condition	attrs	brand	color	model
22	Lexus RX 2016 Black	Cars	Mombasa CBD	Mombasa	Foreign Used	No faults	Lexus	Black	RX
265	New Hyundai Palisade 2021 White	Cars	Mombasa Road	Nairobi	Brand New	No faults	Hyundai	White	Palisade
224	Toyota Hilux 2016 Black	Cars	Mombasa CBD	Mombasa	Foreign Used	First registration	Toyota	Black	Hilux
156	Toyota Land Cruiser 2010 4.6 V8 ZX Black	Cars	Runda	Nairobi	Foreign Used	No faults	Toyota	Black	Land Cruiser
249	Toyota Land Cruiser 2014 4.6 V8 ZX Black	Cars	Karen	Nairobi	Foreign Used	Unpainted, Original parts, No faults	Toyota	Black	Land Cruiser
0	Toyota Land Cruiser Prado 2016 Black	Cars	Mvita	Mombasa	Foreign Used	First registration, No faults	Toyota	Black	Land Cruiser Prado
53	Toyota Land Cruiser Prado 2015 2.7 VVT-i Brown	Cars	Mvita	Mombasa	Foreign Used	No faults	Toyota	Brown	Land Cruiser Prado
241	BMW X5 2015 White	Cars	Mombasa CBD	Mombasa	Foreign Used	First registration	BMW	White	X5
177	Toyota Land Cruiser Prado 2014 Blue	Cars	Nairobi Central	Nairobi	Foreign Used	First owner, Unpainted, Original parts	Toyota	Blue	Land Cruiser Prado
8	BMW X4 2015 xDrive35i Black	Cars	Mombasa CBD	Mombasa	Foreign Used	No faults	BMW	Black	X4

To get only the titles

```
In [10]: # top 10 vehicles with highest price
cars_df.nlargest(10, 'price')[['title', 'category', 'price']]
```

```
Out[10]:
```

	title	category	price
22	Lexus RX 2016 Black	Cars	14500000
265	New Hyundai Palisade 2021 White	Cars	9500000
224	Toyota Hilux 2016 Black	Cars	9000000
156	Toyota Land Cruiser 2010 4.6 V8 ZX Black	Cars	8799999
249	Toyota Land Cruiser 2014 4.6 V8 ZX Black	Cars	8199999
0	Toyota Land Cruiser Prado 2016 Black	Cars	6500000
53	Toyota Land Cruiser Prado 2015 2.7 VVT-i Brown	Cars	6500000
241	BMW X5 2015 White	Cars	6300000
177	Toyota Land Cruiser Prado 2014 Blue	Cars	6150000
8	BMW X4 2015 xDrive35i Black	Cars	5800000

```
In [11]: df["category"]=="Buses & Microbuses"
```

```
Out[11]:
```

0	False
1	False
2	True
3	False
4	False
...	
295	False
296	False
297	False
298	False
299	False

Name: category, Length: 300, dtype: bool

```
In [12]: mask=df["category"]=="Buses & Microbuses"
```

```
In [15]: buses_df=df[mask].copy()
buses_df.head()
```

Out[15]:

	title	category	region	parent_region	condition	attrs	brand	color	model	year
2	Clean NV300 Caravan 2014 Model Dielsel 16 Seater	Buses & Microbuses	Kilimani	Nairobi	Foreign Used	Nissan	Nissan	NaN	Caravan (Urvan)	2014.
25	Nissan Van	Buses & Microbuses	Machakos Town	Machakos	Kenyan Used	Nissan	Nissan	NaN	Caravan (Urvan)	2013.
26	Nissan NV200 2015 White	Buses & Microbuses	Changamwe	Mombasa	Foreign Used	Nissan	Nissan	NaN	NV200	2015.
34	Toyota Hiace Box Petrol on Sale	Buses & Microbuses	Nairobi Central	Nairobi	Kenyan Used	Toyota	Toyota	Grey	HiAce	2008.
55	Chopper 9L Hiace	Buses & Microbuses	Tudor	Mombasa	Foreign Used	Toyota	Toyota	NaN	Grand HiAce	2015.

```
In [17]: # top 5 buses and microbuses with highest price
buses_df.nlargest(5, 'price')
```

Out[17]:

	title	category	region	parent_region	condition	attrs	brand	color	model	year
148	Mazda Bongo	Buses & Microbuses	Ridgeways	Nairobi	Foreign Used	Unpainted, No faults	Mazda	White	Bongor	
221	Selling Buses In Mombasa Town	Buses & Microbuses	Tudor	Mombasa	Foreign Used	Mitsubishi	Mitsubishi	NaN	Fusion Ros	
174	Roller Coaster	Buses & Microbuses	Nairobi Central	Nairobi	Foreign Used	Toyota	Toyota	NaN	Coaster	
211	Toyota Coaster 2014 White	Buses & Microbuses	Mombasa CBD	Mombasa	Foreign Used	Toyota	Toyota	NaN	Coaster	
268	Toyota Hiace 2015 White	Buses & Microbuses	Ziwa la Ngombe	Nyali	Brand New	Toyota	Toyota	NaN	HiAce	

```
In [19]: # top 5 buses and microbuses with highest price
buses_df.nlargest(5, 'price')[['title', 'category', 'price']]
```

Out[19]:

	title	category	price
148	Mazda Bongo	Buses & Microbuses	11200000
221	Selling Buses In Mombasa Town	Buses & Microbuses	5200000
174	Roller Coaster	Buses & Microbuses	4900000
211	Toyota Coaster 2014 White	Buses & Microbuses	4300000
268	Toyota Hiace 2015 White	Buses & Microbuses	3800000

In [21]: `df["category"]=="Trucks & Trailers"`

Out[21]:

```

0    False
1    False
2    False
3    False
4    False
...
295  False
296    True
297    True
298  False
299  False
Name: category, Length: 300, dtype: bool

```

In []: `mask=df["category"]=="Trucks & Trailers"`In [22]: `trucks_df=df[mask].copy()
trucks_df.head()`

Out[22]:

	title	category	region	parent_region	condition	attrs	brand	color	model	yor
2	Clean NV300 Caravan 2014 Model Dielsel 16 Seater	Buses & Microbuses	Kilimani	Nairobi	Foreign Used	Nissan	Nissan	NaN	Caravan (Urvan)	2014.
25	Nissan Van	Buses & Microbuses	Machakos Town	Machakos	Kenyan Used	Nissan	Nissan	NaN	Caravan (Urvan)	2013.
26	Nissan NV200 2015 White	Buses & Microbuses	Changamwe	Mombasa	Foreign Used	Nissan	Nissan	NaN	NV200	2015.
34	Toyota Hiace Box Petrol on Sale	Buses & Microbuses	Nairobi Central	Nairobi	Kenyan Used	Toyota	Toyota	Grey	HiAce	2008.
55	Chopper 9L Hiace	Buses & Microbuses	Tudor	Mombasa	Foreign Used	Toyota	Toyota	NaN	Grand HiAce	2015.

The above output is what the question is asking for. So take a screenshot.

```
In [23]: trucks_df.nlargest(5,"price")
```

```
Out[23]:
```

	title	category	region	parent_region	condition	attrs	brand	color	mode
148	Mazda Bongo	Buses & Microbuses	Ridgeways	Nairobi	Foreign Used	Unpainted, No faults	Mazda	White	Bong
221	Selling Buses In Mombasa Town	Buses & Microbuses	Tudor	Mombasa	Foreign Used	Mitsubishi	Mitsubishi	NaN	Fusi Ros
174	Roller Coaster	Buses & Microbuses	Nairobi Central	Nairobi	Foreign Used	Toyota	Toyota	NaN	Coaste
211	Toyota Coaster 2014 White	Buses & Microbuses	Mombasa CBD	Mombasa	Foreign Used	Toyota	Toyota	NaN	Coaste
268	Toyota Hiace 2015 White	Buses & Microbuses	Ziwa la Ngombe	Nyali	Brand New	Toyota	Toyota	NaN	HiAc

```
In [25]: trucks_df.nlargest(5,"price")[["category","title","price"]]
```

```
Out[25]:
```

	category	title	price
148	Buses & Microbuses	Mazda Bongo	11200000
221	Buses & Microbuses	Selling Buses In Mombasa Town	5200000
174	Buses & Microbuses	Roller Coaster	4900000
211	Buses & Microbuses	Toyota Coaster 2014 White	4300000
268	Buses & Microbuses	Toyota Hiace 2015 White	3800000

Plotting

I will demonstrate how to solve the plotting challenge using the following question:

Use **matplotlib** to come up with a plot indicating the **top 5 regions** that we have in the vehicle_dataset

```
In [27]: # get number of rows with vehicle brands
df['brand'].value_counts()
```

```
Out[27]: Toyota      82
         Nissan      33
         Mitsubishi  32
         Mazda       26
         Subaru      22
         Volkswagen  21
         Isuzu       19
         Honda       17
         BMW         17
         Mercedes-Benz 9
         Suzuki      5
         Lexus       4
         Tata        2
         Volvo       2
         Ashok Leyland 2
         Land Rover  1
         Shacman     1
         Other       1
         Hyundai    1
         Daihatsu    1
         Audi        1
         Name: brand, dtype: int64
```

```
In [28]: # grab the top 10
         df['brand'].value_counts()[:10]
```

```
Out[28]: Toyota      82
         Nissan      33
         Mitsubishi  32
         Mazda       26
         Subaru      22
         Volkswagen  21
         Isuzu       19
         Honda       17
         BMW         17
         Mercedes-Benz 9
         Name: brand, dtype: int64
```

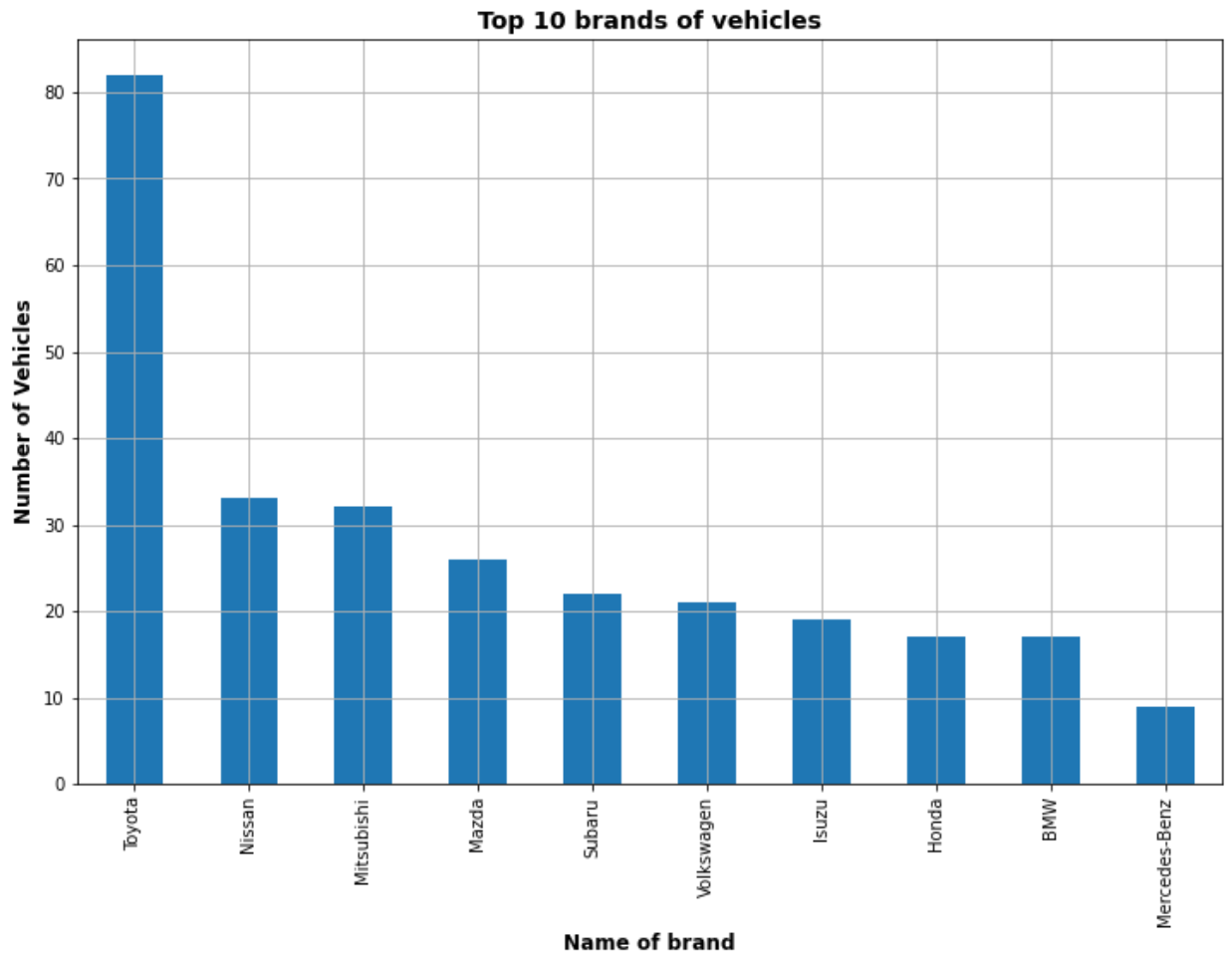
```
In [30]: # make it a variable
         top_10 = df['brand'].value_counts()[:10]
```

Now to create a bar plot of the top 5 regions

```
In [32]: plt.figure(figsize=(12,8))
         plt.title("Top 10 brands of vehicles", fontsize=14, fontweight='bold')
         top_10.plot.bar()
         plt.xlabel('Name of brand',fontsize=12, fontweight='bold')
         plt.ylabel('Number of Vehicles',fontsize=12, fontweight='bold')
         plt.grid()

         # save the plot to file
         fig = plt.gcf()
         fig.savefig('top-10-brands.png')

         plt.show()
```



In []: