

IMF

By Gerard Utoware

Table of Contents

Overviews – Page 3

Q1 – Page 4

Overview of the tool used

The main advantage of python in comparison to other data analytic tools such as excel, would be Python's ability to handle large volumes of data without hindering productivity. Running scripts using the libraries on allows for more automation and ease of obtaining analysed data. With this task, the data was imported from python which ensures that data will not be lost or tampered with while performing data analysis. After defining the columns from the imported data, it was less tedious to run scripts than to perform individual analysis on excel.

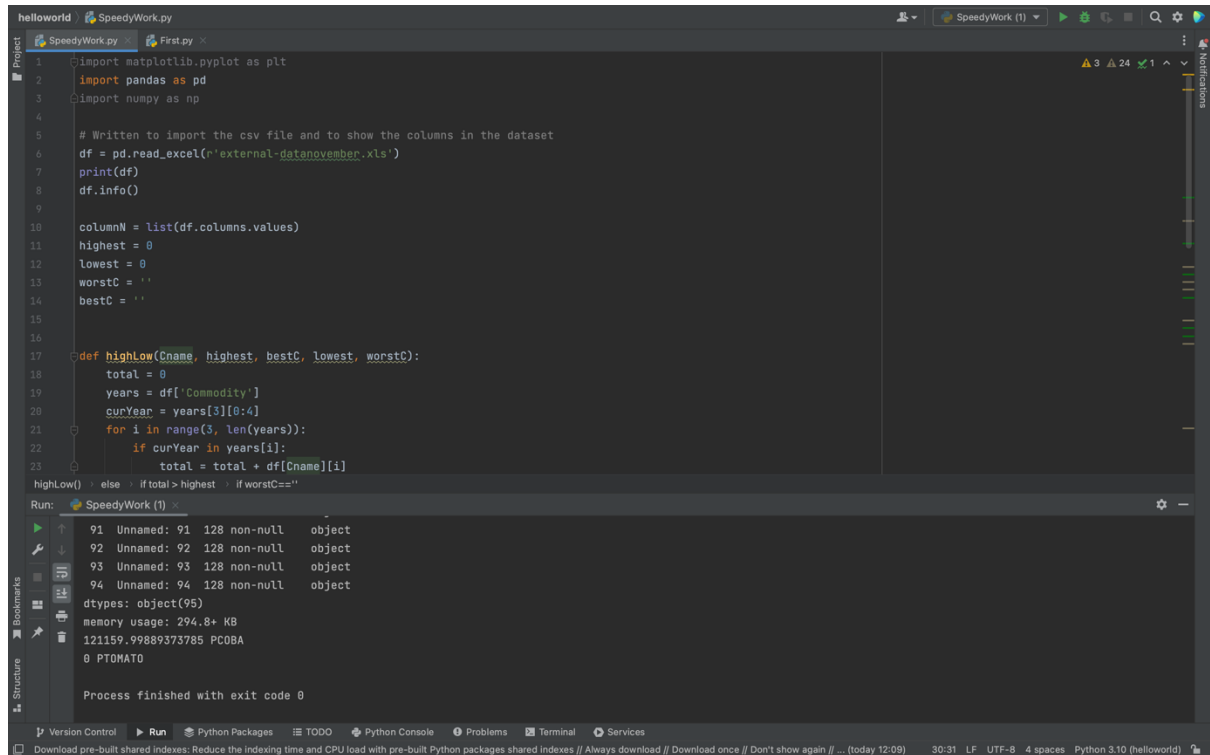
Overview of the libraries used

Numpy, pandas and matplotlib were used in the script. All three python libraries allow different functions to be performed. Pandas allows data analysis scripts to be conducted on python. It utilises two other libraries, being numpy and matplotlib. Matplotlib allows for data visualisation through a variety of charts such as bar and pie charts. Numpy permits mathematical operations in python. The combination of these three libraries is what allowed the following data and charts to be formed and analysed. In addition to these, seaborn was installed in order for a heatmap to be printed and shown.

Overview of the dataset

The dataset imported onto python tallied the prices of an array of commodities from the years 1990 to the current day. Certain commodities were not recorded at the start of the dataset time period, such as 'food and beverage', 'industrial input and 'agriculture price index'. There were a few years where the index was 0 for different industries such as PTOMATO, which was the lowest industry in the dataset. By contrast, the industry with the highest is PCOBA. The dataset, however, does not take into consideration external factors that affect the supply and demand of these commodities throughout the years. For example, overall weather by the month and year can severely affect the commodity price of agriculture. Transportation and geopolitics are another major factor that has an effect on the price of commodity. With that being said, higher priced indexes are still more valuable and wanted more so than lower commodities.

Q1&2



The screenshot shows a VS Code editor with a file named `SpeedyWork.py` open. The code imports `matplotlib.pyplot` as `plt`, `pandas` as `pd`, and `numpy` as `np`. It reads an Excel file `external-datanovember.xls` into a DataFrame `df` and prints its information. A function `highLow` is defined, which takes a column name and four comparison values. It iterates over the first three years of the 'Commodity' column and updates a total based on the comparison. The `highLow()` function is called at the end of the script.

```
1 import matplotlib.pyplot as plt
2 import pandas as pd
3 import numpy as np
4
5 # Written to import the csv file and to show the columns in the dataset
6 df = pd.read_excel('external-datanovember.xls')
7 print(df)
8 df.info()
9
10 columnN = list(df.columns.values)
11 highest = 0
12 lowest = 0
13 worstC = ''
14 bestC = ''
15
16
17 def highLow(Cname, highest, bestC, lowest, worstC):
18     total = 0
19     years = df['Commodity']
20     curYear = years[3][0:4]
21     for i in range(3, len(years)):
22         if curYear in years[i]:
23             total = total + df[Cname][i]
24
25 highLow() # else : if total > highest : if worstC==''
```

The Run console shows the output of the script:

```
91 Unnamed: 91 128 non-null object
92 Unnamed: 92 128 non-null object
93 Unnamed: 93 128 non-null object
94 Unnamed: 94 128 non-null object
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PC08A
0 PTOMATO

Process finished with exit code 0
```

The image displays two screenshots of a VS Code editor window. The top screenshot shows a Python script named 'SpeedyWork.py' with a function 'highLow()' that iterates through a list of years and calculates the highest and lowest values for a specific commodity. The bottom screenshot shows the same script with additional code to print the results and a small inset window showing a terminal output.

```
21 for i in range(3, len(years)):
22     if curYear in years[i]:
23         total = total + df[Cname][i]
24     else:
25         if total > highest:
26             highest = total
27             bestC = Cname
28             if worstC == '':
29                 lowest = total
30                 worstC = Cname
31         else:
32             if total < lowest:
33                 lowest = total
34                 worstC = Cname
35                 total = df[Cname][i]
36                 curYear = years[i][0:4]
37
38 return highest, bestC, lowest, worstC
39
40 columnN.remove('Commodity')
41 df = df.iloc[3:]
42 for x in columnN:
43     highLow()
44     if total > highest:
45         if worstC == ''
```

Run: SpeedyWork (1) x

```
91 Unnamed: 91 128 non-null object
92 Unnamed: 92 128 non-null object
93 Unnamed: 93 128 non-null object
94 Unnamed: 94 128 non-null object
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO
Process finished with exit code 0
```

Version Control Run Python Packages TODO Python Console Problems Terminal Services

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // ... (today 12:09) 30:31 LF UTF-8 4 spaces Python 3.10 (helloworld)

```
30     worstC = Cname
31 else:
32     if total < lowest:
33         lowest = total
34         worstC = Cname
35         total = df[Cname][i]
36         curYear = years[i][0:4]
37
38 return highest, bestC, lowest, worstC
39
40 columnN.remove('Commodity')
41 df = df.iloc[3:]
42 for x in columnN:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44
45 print(highest, bestC)
46 print(lowest, worstC)
47
```

highLow() else if total > highest if worstC == ''

Run: SpeedyWork (1) x

```
91 Unnamed: 91 128 non-null object
92 Unnamed: 92 128 non-null object
93 Unnamed: 93 128 non-null object
94 Unnamed: 94 128 non-null object
dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO
Process finished with exit code 0
```

Version Control Run Python Packages TODO Python Console Problems Terminal Services

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // ... (today 12:09) 30:31 LF UTF-8 4 spaces Python 3.10 (helloworld)

Highest index was the PCOBA at 1211.998893773785. The lowest index was PTOMATO with 0.

Q3,4&5

```

30     worstC = Cname
31 else:
32     if total < lowest:
33         lowest = total
34         worstC = Cname
35         total = df[Cname][i]
36         curYear = years[i][0:4]
37
38 return highest, bestC, lowest, worstC
39
40
41 columnN.remove('Commodity')
42 df = df.iloc[3:]
43 for x in columnN:
44     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
45
46 print(highest, bestC)
47 print(lowest, worstC)
48
49 print(df[['PCOAL', 'PCOPP', 'PRUBB']].mean())

```

Run: SpeedyWork (1) x

```

dtypes: object(95)
memory usage: 294.8+ KB
121159.99889373785 PCOBA
0 PTOMATO
PCOAL    100.144245
PCOPP    4631.063670
PRUBB     75.988382
dtype: float64
Process finished with exit code 0

```

The average of the 3 respective columns to get the average price index for coal, copper and rubber. Average of coal, copper and rubber are 100.14, 4631.06 and 75.99.

Q6&7

```

31     lowest = total
32     worstC = Cname
33
34
35 return highest, bestC, lowest, worstC
36
37
38 columnN.remove('Commodity')
39 df = df.iloc[3:]
40 namescoff=['PCOFFROB', 'PCOFFOTM']
41 for x in namescoff:
42     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
43     print(highest, bestC)
44
45 namesoil=['POILAPSP', 'POILBRE', 'POILOUB', 'POILWTI']
46 highest = 0
47 lowest = 0
48 for x in namesoil:
49     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
50     print(highest, bestC)
51
52 print(df[['PCOAL', 'PCOPP', 'PRUBB']].mean())

```

Run: SpeedyWork (1) x

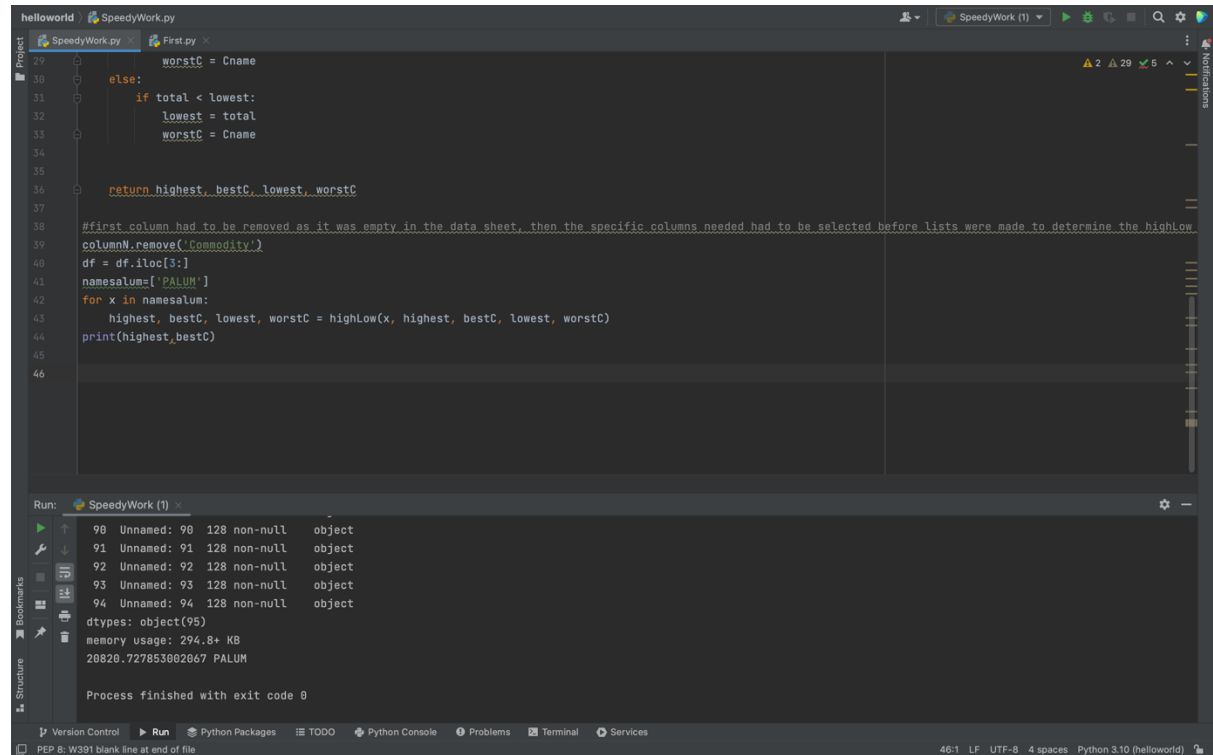
```

dtypes: object(95)
memory usage: 294.8+ KB
1865.4258743647658 PCOFFOTM
1692.2381265068614 POILAPSP
PCOAL    100.144245
PCOPP    4631.063670
PRUBB     75.988382
dtype: float64
Process finished with exit code 0

```

By defining two different lists for the columns containing different coffee and crude oil, the best type for each respective list. The best coffee for 2022 was PCOFFOTM with 1865.43(2dp). The best oil was POILAPSP with 1692.24(2dp).

Further Data



```
29     worstC = Cname
30
31     else:
32         if total < lowest:
33             lowest = total
34             worstC = Cname
35
36     return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesalum=['PALUM']
42 for x in namesalum:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44 print(highest,bestC)
45
46
```

Run: SpeedyWork (1) x

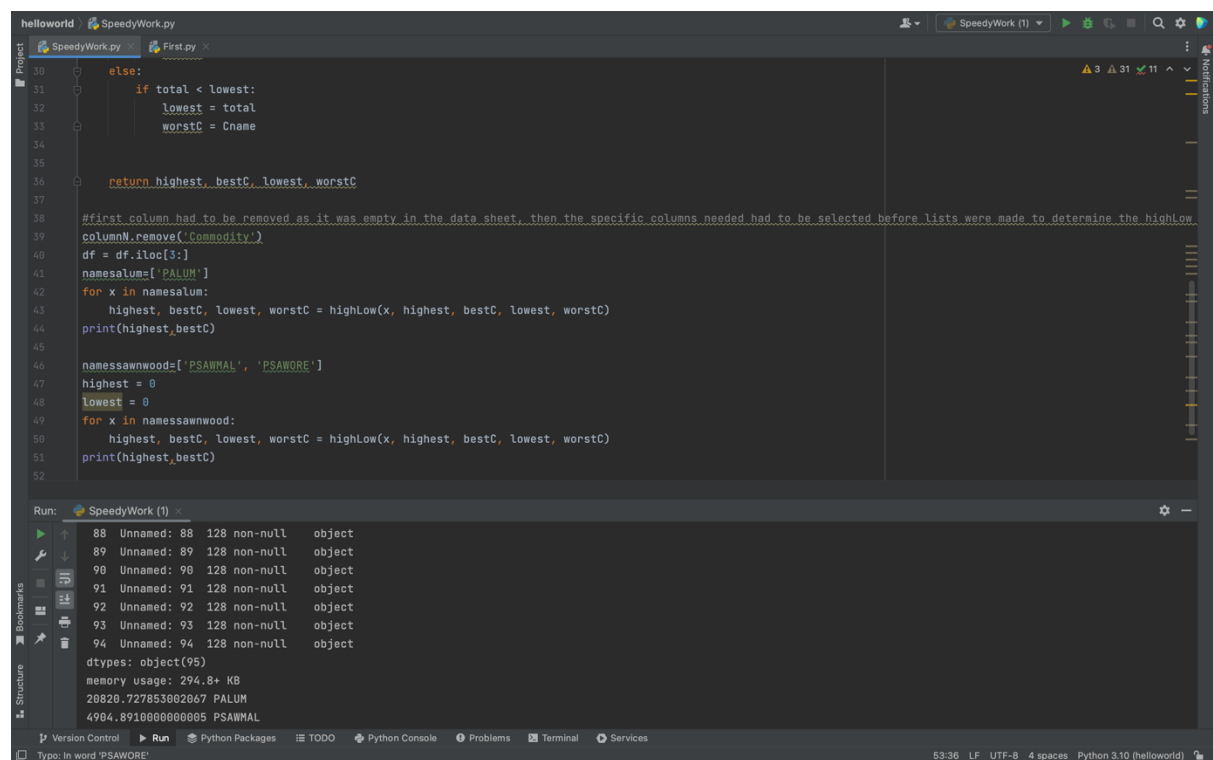
```
90 Unnamed: 90 128 non-null object
91 Unnamed: 91 128 non-null object
92 Unnamed: 92 128 non-null object
93 Unnamed: 93 128 non-null object
94 Unnamed: 94 128 non-null object
dtypes: object(95)
memory usage: 294.8+ KB
20820.727853002067 PALUM
Process finished with exit code 0
```

Version Control | Run | Python Packages | TODO | Python Console | Problems | Terminal | Services

PEP 8: W391 blank line at end of file

46:1 LF UTF-8 4 spaces Python 3.10 (helloworld)

The best financial month for Aluminium had an index value of 20820.73(2dp).



```
30     else:
31         if total < lowest:
32             lowest = total
33             worstC = Cname
34
35     return highest, bestC, lowest, worstC
36
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesalum=['PALUM']
42 for x in namesalum:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44 print(highest,bestC)
45
46 namesawnwood=['PSAWMAL', 'PSAWORE']
47 highest = 0
48 lowest = 0
49 for x in namesawnwood:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51 print(highest,bestC)
52
```

Run: SpeedyWork (1) x

```
88 Unnamed: 88 128 non-null object
89 Unnamed: 89 128 non-null object
90 Unnamed: 90 128 non-null object
91 Unnamed: 91 128 non-null object
92 Unnamed: 92 128 non-null object
93 Unnamed: 93 128 non-null object
94 Unnamed: 94 128 non-null object
dtypes: object(95)
memory usage: 294.8+ KB
20820.727853002067 PALUM
4904.8910000000005 PSAWMAL
Type: In word 'PSAWORE'
```

Version Control | Run | Python Packages | TODO | Python Console | Problems | Terminal | Services

53:36 LF UTF-8 4 spaces Python 3.10 (helloworld)

When looking at sawnwood, Hard sawnwood had the largest value out of both hard and soft sawnwood, with a value of 4904.89(2dp).

```

36     return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesalum=['PALUN']
42 for x in namesalum:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44 print(highest,bestC)
45
46 namesawnwood=['PSAWMAL', 'PSAWORE']
47 highest = 0
48 lowest = 0
49 for x in namesawnwood:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51 print(highest,bestC)
52
53 print(df[['PALUN', 'PSAWMAL', 'PSAWORE']].mean())

```

Run: SpeedyWork (1)

```

dtypes: object(95)
memory usage: 294.8+ KB
20820.727853002067 PALUN
4904.8910000000005 PSAWMAL
PALUN    1804.348573
PSAWMAL    707.637750
PSAWORE    307.313475
dtype: float64
Process finished with exit code 0

```

The average of all the values in their respective columns, this helps to confirm that hard sawnwood is more valuable than soft sawnwood.

```

34
35
36     return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
42 for x in namesgas:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44 print(highest,bestC)
45
46 nameswool=['PWOLLC', 'PWOLLE']
47 highest = 0
48 lowest = 0
49 for x in nameswool:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51 print(highest,bestC)
52
53 print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PWOLLC', 'PWOLLE']].mean())

```

Run: SpeedyWork (1)

```

87 PAPPLE    397 non-null    object
88 Unnamed: 88    128 non-null    object
89 Unnamed: 89    128 non-null    object
90 Unnamed: 90    128 non-null    object
91 Unnamed: 91    128 non-null    object
92 Unnamed: 92    128 non-null    object
93 Unnamed: 93    128 non-null    object
94 Unnamed: 94    128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
3367.109502740262 PNGAS

```

Highest value natural gas, being 3367.11(2dp).


```

31 if total < lowest:
32     lowest = total
33     worstC = Cname
34
35
36 return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
42 for x in namesgas:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44 print(highest, bestC)
45
46 nameswool=['PWOLLC', 'PWOLLE']
47 highest = 0
48 lowest = 0
49 for x in nameswool:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51 print(highest, bestC)
52
53 print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PWOLLC', 'PWOLLE']].mean())

```

Run: SpeedyWork (1)

```

88 Unnamed: 88    128 non-null    object
89 Unnamed: 89    128 non-null    object
90 Unnamed: 90    128 non-null    object
91 Unnamed: 91    128 non-null    object
92 Unnamed: 92    128 non-null    object
93 Unnamed: 93    128 non-null    object
94 Unnamed: 94    128 non-null    object
dtypes: object(95)
memory usage: 294.8+ KB
3367.189582740262 PNGAS
8576.9954975 PWOLLE

```

Fine wool has the highest value in the spreadsheet when compared to coarse wool. Highest value is 8576.9954975

```

33     worstC = Cname
34
35
36 return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
42 for x in namesgas:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44 print(highest, bestC)
45
46 nameswool=['PWOLLC', 'PWOLLE']
47 highest = 0
48 lowest = 0
49 for x in nameswool:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51 print(highest, bestC)
52
53 print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PWOLLC', 'PWOLLE']].mean())

```

Run: SpeedyWork (1)

```

8576.9954975 PWOLLE
PNGAS      142.559089
PNGASEU     6.684325
PNGASJP     8.614802
PNGASUS     3.806763
PWOLLC     706.334417
PWOLLE     927.112009
dtype: float64
Process finished with exit code 0

```

Average of their respective columns. Fine wool still has a higher average than its coarse counterpart.

```

32 lowest = total
33 worstC = Cname
34
35
36 return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
42 for x in namesgas:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44     print(highest,bestC)
45
46 nameswool=['PW00LC', 'PW00LF']
47 highest = 0
48 lowest = 0
49 for x in nameswool:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51     print(highest,bestC)
52
53 print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PW00LC', 'PW00LF']].median())

```

Run: SpeedyWork (1) ×

```

3367.189582740262 PNGAS
8576.9954975 PW00LF
PNGAS 123.390647
PNGASEU 4.268080
PNGASJP 6.451739
PNGASUS 3.033382
PW00LC 607.981099
PW00LF 853.430829
dtype: float64

```

Process finished with exit code 0

Median of their respective columns.

```

33 lowest = total
34 worstC = Cname
35
36 return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
42 for x in namesgas:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44     print(highest,bestC)
45
46 nameswool=['PW00LC', 'PW00LF']
47 highest = 0
48 lowest = 0
49 for x in nameswool:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51     print(highest,bestC)
52
53 print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PW00LC', 'PW00LF']].mode())

```

Run: SpeedyWork (1) ×

```

93 Unnamed: 93 120 non-null object
94 Unnamed: 94 128 non-null object
dtypes: object(95)
memory usage: 294.8+ KB
3367.189582740262 PNGAS
8576.9954975 PW00LF

```

	PNGAS	PNGASEU	PNGASJP	PNGASUS	PW00LC	PW00LF
0	43.928331	2.69	3.5	1.1485	328.23938	596.018418
1	43.994589	NaN	3.82	1.18	418.442568	NaN
2	45.856918	NaN	NaN	1.251167	NaN	NaN
3	48.414884	NaN	NaN	1.26	NaN	NaN
4	48.514552	NaN	NaN	1.3028	NaN	NaN

Top 5 most common values in each column in the dataset. Only PNGAS has 5 different unique values. Other columns have 'NaN' which means that its Not A Number.

```

helloworld SpeedyWork.py
Project SpeedyWork.py First.py
33 worstC = Cname
34
35
36 return highest, bestC, lowest, worstC
37
38 #first column had to be removed as it was empty in the data sheet, then the specific columns needed had to be selected before lists were made to determine the highLow
39 columnN.remove('Commodity')
40 df = df.iloc[3:]
41 namesgas=['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS']
42 for x in namesgas:
43     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
44     print(highest, bestC)
45
46 nameswool=['PW00LC', 'PW00LF']
47 highest = 0
48 lowest = 0
49 for x in nameswool:
50     highest, bestC, lowest, worstC = highLow(x, highest, bestC, lowest, worstC)
51     print(highest, bestC)
52
53 print(df[['PNGAS', 'PNGASEU', 'PNGASJP', 'PNGASUS', 'PW00LC', 'PW00LF']].std())

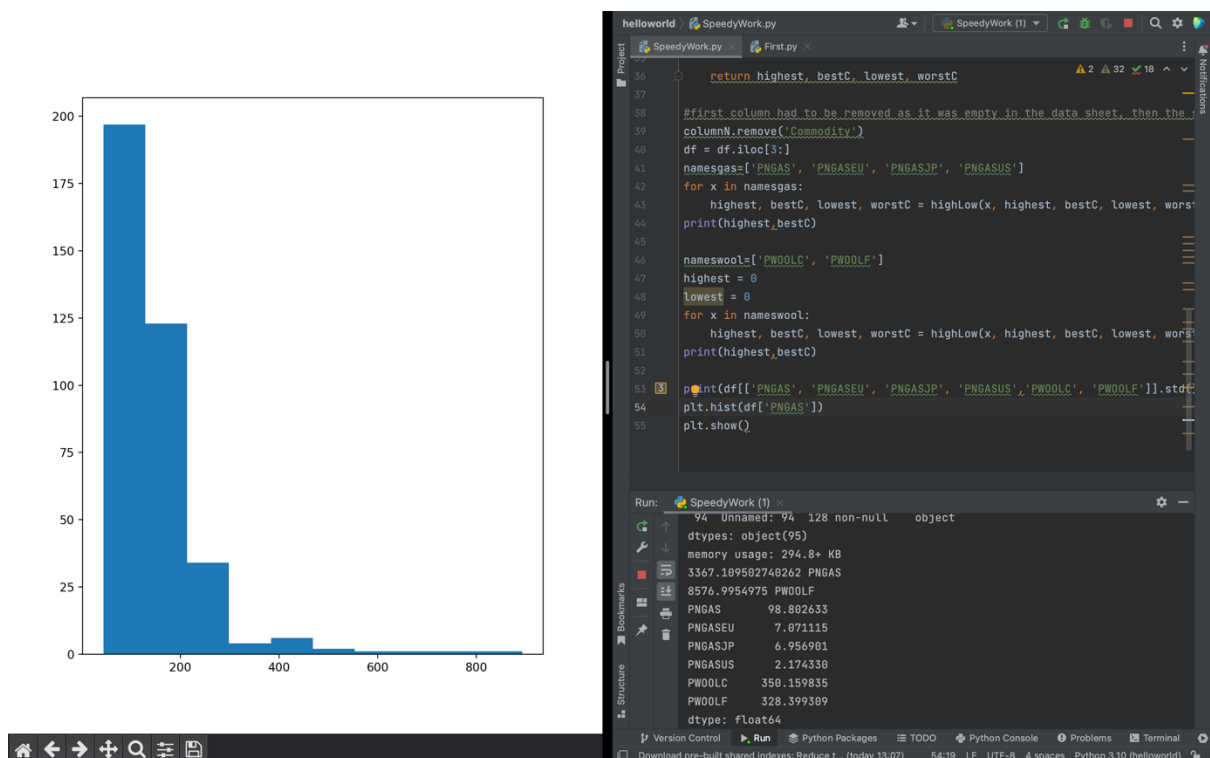
```

```

Run: SpeedyWork (1) x
3367.189582740262 PNGAS
8576.9954975 PW00LF
PNGAS 98.802633
PNGASEU 7.071115
PNGASJP 6.956901
PNGASUS 2.174330
PW00LC 350.159835
PW00LF 328.399309
dtype: float64
Process finished with exit code 0

```

The standard deviation measures the spread of data in each column. The lower the standard deviation, the more the data points are to be found around the mean.



Histogram of the PNGAS column to show where majority of the data points lay, being between 50-200.