ACST 890- TAKE HOME QUIZ 2 SOLUTIONS: 45535531-ROHAN MARCUS DEANS

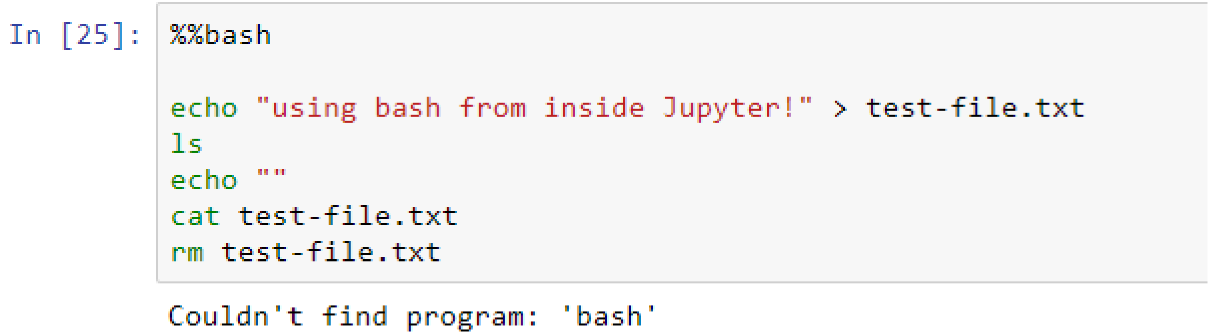
GITHUB Username: 45535531

Repository Name: Quiz1 (https://github.com/45535531/Quiz1.git)

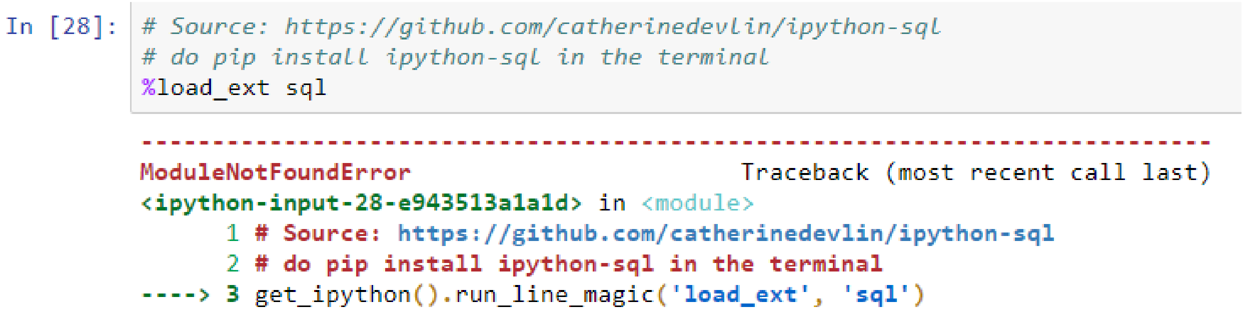
File Name: 45535531DeansRohanMarcusTHQ2.pdf

Other Attached files on GITHUB: codings.txt

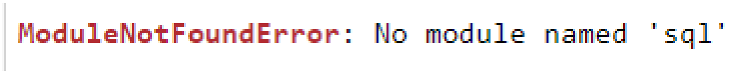
1.a.



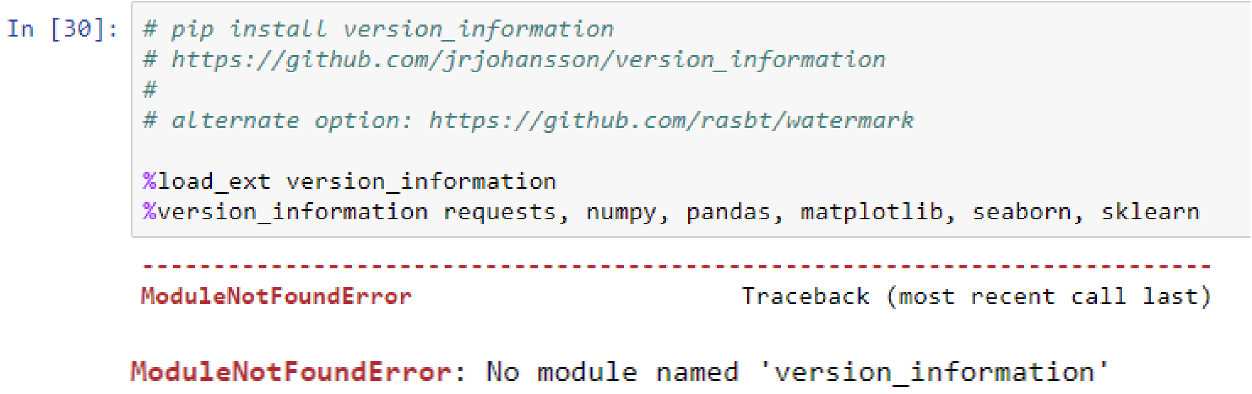
```
In [25]: %%bash

         echo "using bash from inside Jupyter!" > test-file.txt
         ls
         echo ""
         cat test-file.txt
         rm test-file.txt

         Couldn't find program: 'bash'
```



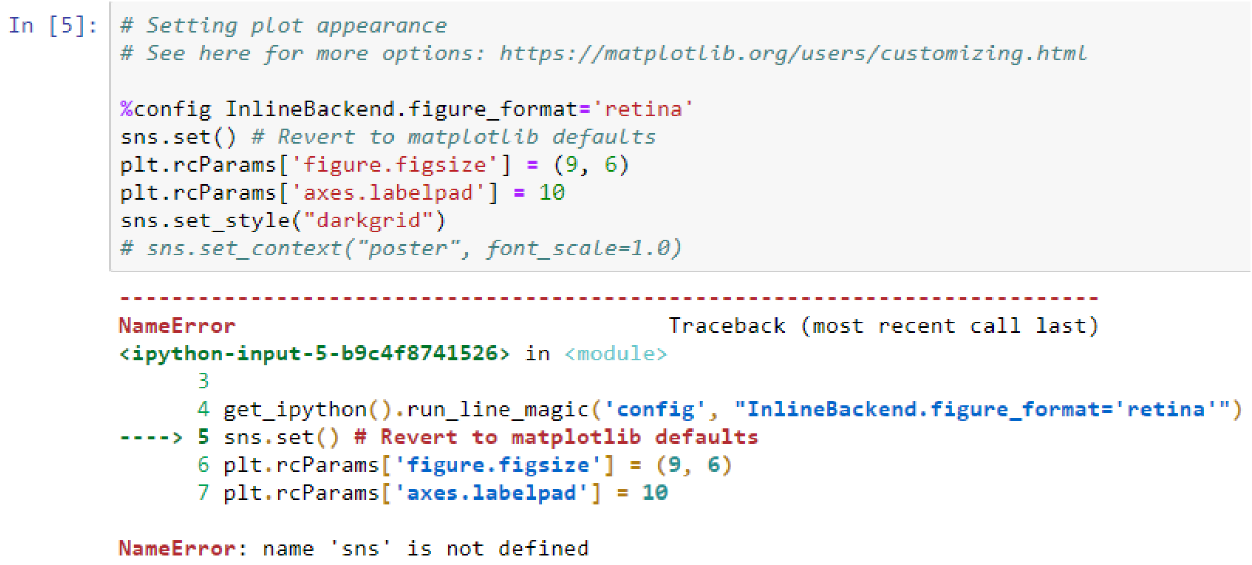
```
In [28]: # Source: https://github.com/catherinedevlin/ipython-sql
         # do pip install ipython-sql in the terminal
         %load_ext sql

         ---------------------------------------------------------------------------
         ModuleNotFoundError                       Traceback (most recent call last)
         <ipython-input-28-e943513a1a1d> in <module>
               1 # Source: https://github.com/catherinedevlin/ipython-sql
               2 # do pip install ipython-sql in the terminal
         ----> 3 get_ipython().run_line_magic('load_ext', 'sql')
```



```
ModuleNotFoundError: No module named 'sql'
```

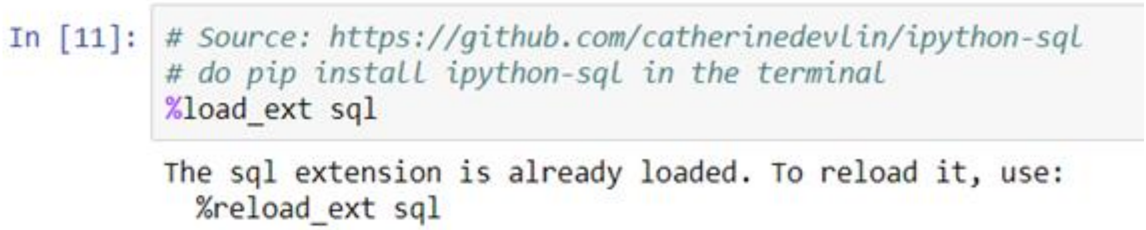


```
In [30]: # pip install version_information
         # https://github.com/jrjohansson/version_information
         #
         # alternate option: https://github.com/rasbt/watermark

         %load_ext version_information
         %version_information requests, numpy, pandas, matplotlib, seaborn, sklearn

         ---------------------------------------------------------------------------
         ModuleNotFoundError                       Traceback (most recent call last)

         ModuleNotFoundError: No module named 'version_information'
```

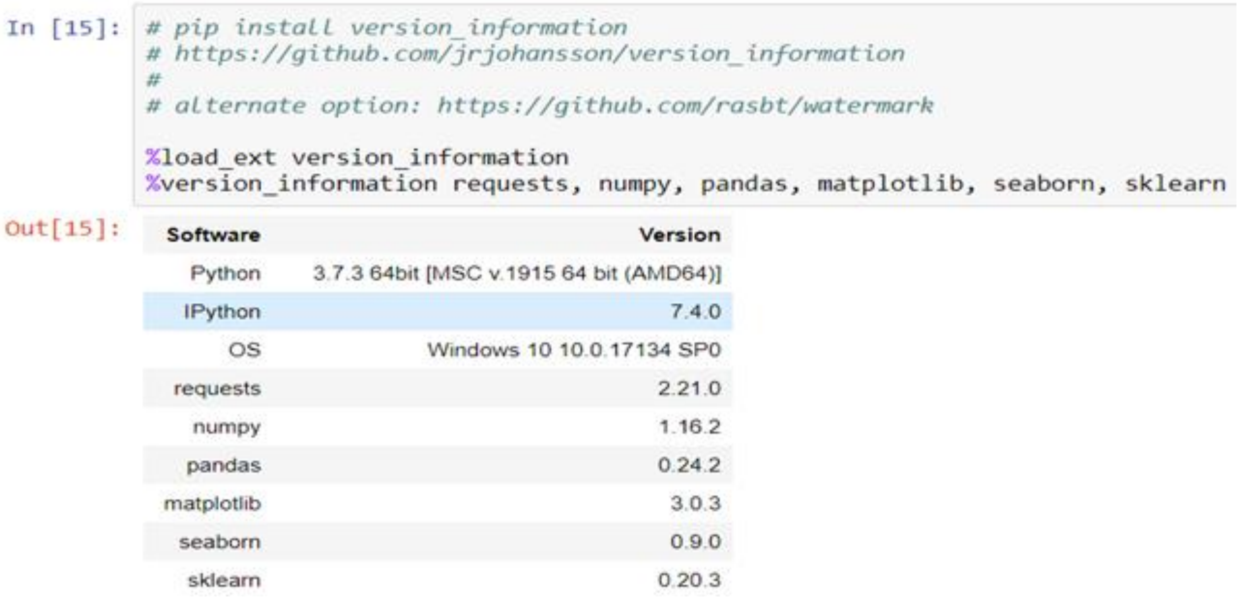


```
In [5]: # Setting plot appearance
        # See here for more options: https://matplotlib.org/users/customizing.html

        %config InlineBackend.figure_format='retina'
        sns.set() # Revert to matplotlib defaults
        plt.rcParams['figure.figsize'] = (9, 6)
        plt.rcParams['axes.labelpad'] = 10
        sns.set_style("darkgrid")
        # sns.set_context("poster", font_scale=1.0)

        ---------------------------------------------------------------------------
        NameError                                 Traceback (most recent call last)
        <ipython-input-5-b9c4f8741526> in <module>
              3
              4 get_ipython().run_line_magic('config', "InlineBackend.figure_format='retina'")
        ----> 5 sns.set() # Revert to matplotlib defaults
              6 plt.rcParams['figure.figsize'] = (9, 6)
              7 plt.rcParams['axes.labelpad'] = 10

        NameError: name 'sns' is not defined
```

1.b.



```
In [11]: # Source: https://github.com/catherinedevlin/ipython-sql
         # do pip install ipython-sql in the terminal
         %load_ext sql

         The sql extension is already loaded. To reload it, use:
           %reload_ext sql
```



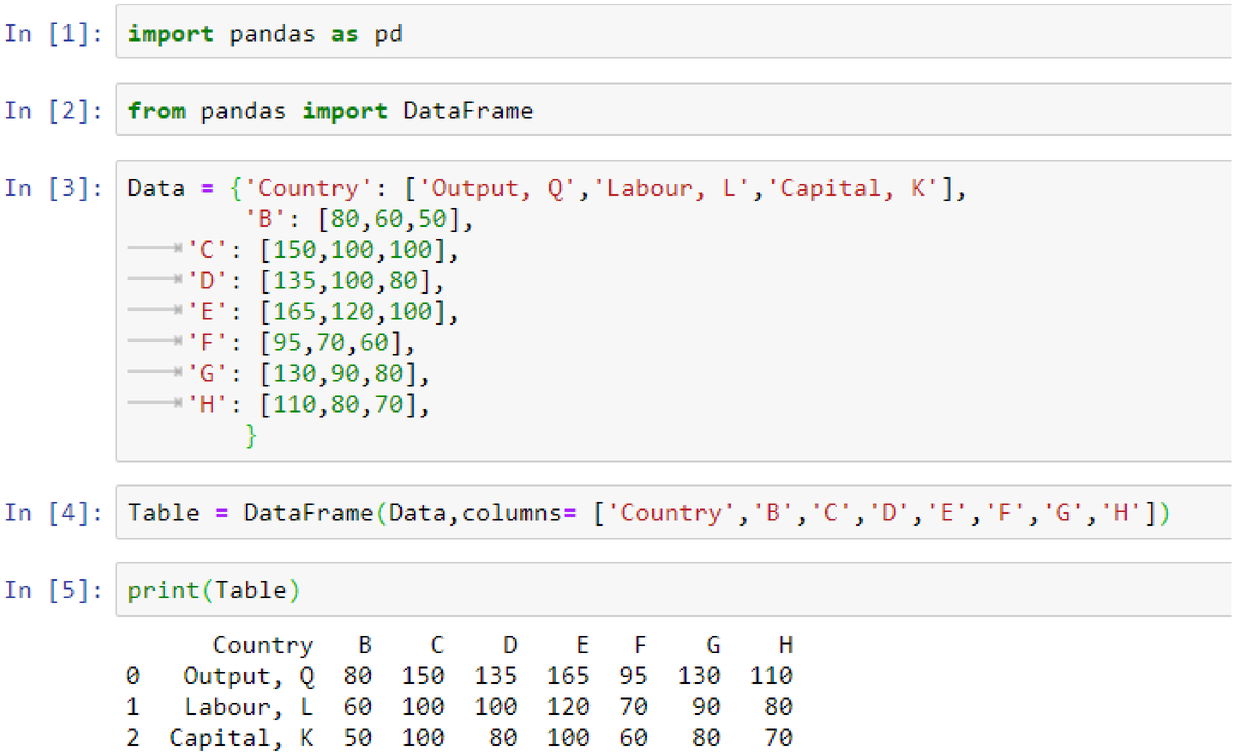
In [15]:

```
# pip install version_information
# https://github.com/jrjohansson/version_information
#
# alternate option: https://github.com/rasbt/watermark

%load_ext version_information
%version_information requests, numpy, pandas, matplotlib, seaborn, sklearn
```

Out[15]:

Software	Version
Python	3.7.3 64bit [MSC v.1915 64 bit (AMD64)]
IPython	7.4.0
OS	Windows 10 10.0.17134 SP0
requests	2.21.0
numpy	1.16.2
pandas	0.24.2
matplotlib	3.0.3
seaborn	0.9.0
sklearn	0.20.3

2.



```
In [1]: import pandas as pd
```

```
In [2]: from pandas import DataFrame
```

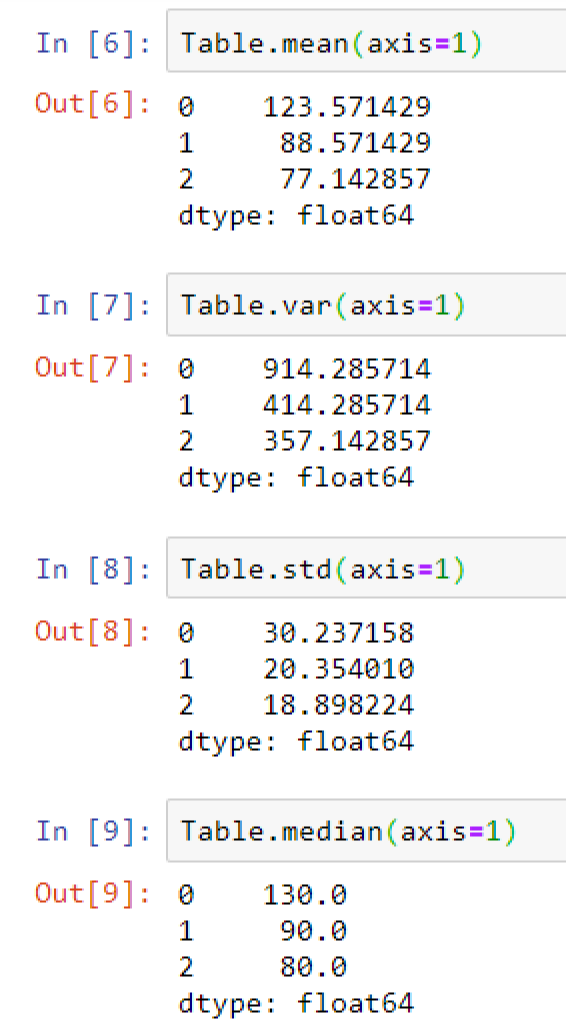
```
In [3]: Data = {'Country': ['Output, Q','Labour, L','Capital, K'],
                'B': [80,60,50],
            'C': [150,100,100],
            'D': [135,100,80],
            'E': [165,120,100],
            'F': [95,70,60],
            'G': [130,90,80],
            'H': [110,80,70],
                }
```

```
In [4]: Table = DataFrame(Data,columns= ['Country','B','C','D','E','F','G','H'])
```

```
In [5]: print(Table)
```

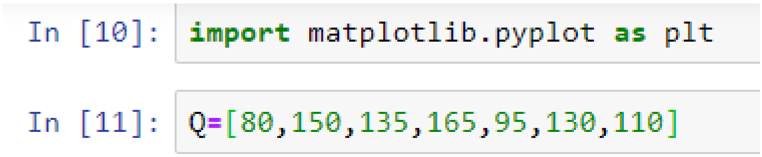
```
       Country   B    C    D    E   F    G    H
0    Output, Q  80  150  135  165  95  130  110
1    Labour, L  60  100  100  120  70   90   80
2   Capital, K  50  100   80  100  60   80   70
```

a.

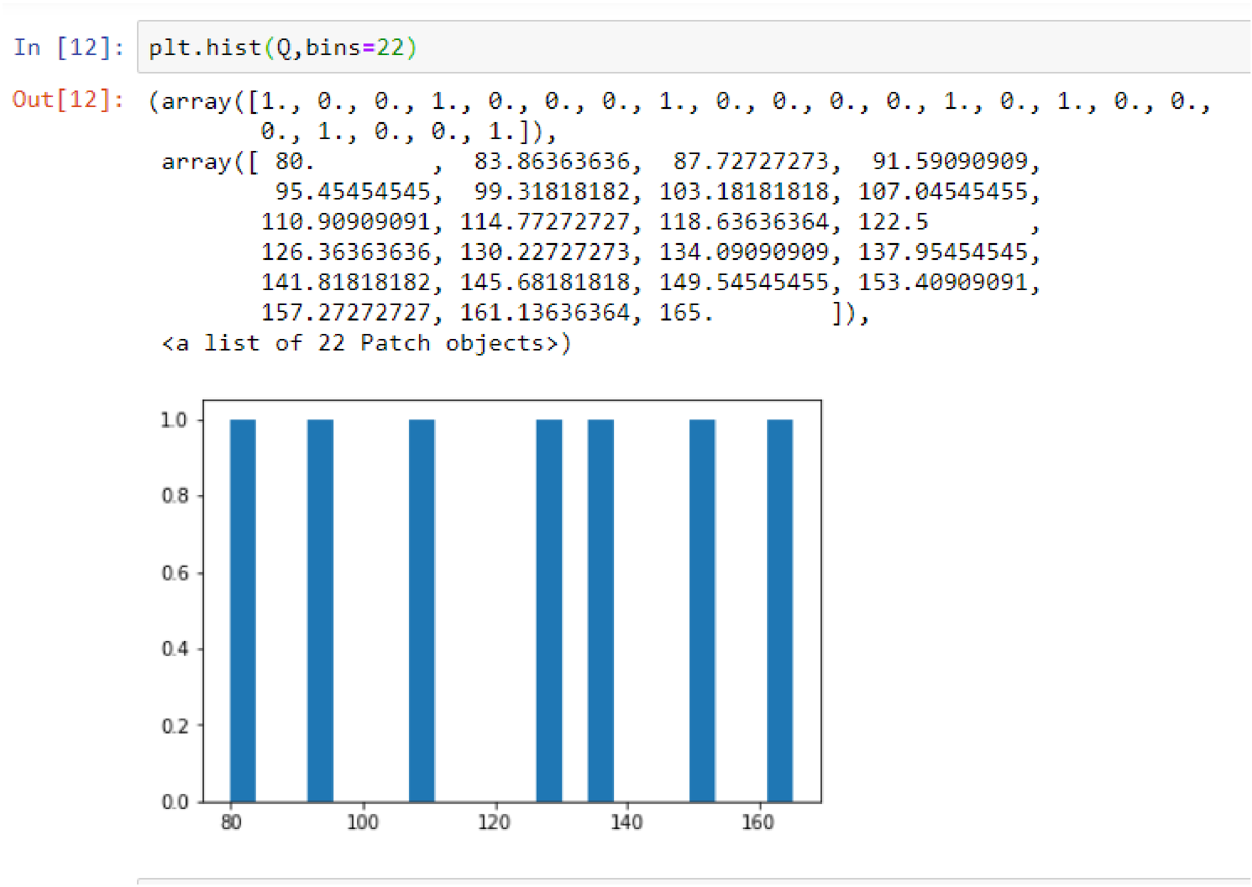


```
In [6]: Table.mean(axis=1)

Out[6]: 0    123.571429
        1     88.571429
        2     77.142857
        dtype: float64

In [7]: Table.var(axis=1)

Out[7]: 0    914.285714
        1    414.285714
        2    357.142857
        dtype: float64

In [8]: Table.std(axis=1)

Out[8]: 0    30.237158
        1    20.354010
        2    18.898224
        dtype: float64

In [9]: Table.median(axis=1)

Out[9]: 0    130.0
        1     90.0
        2     80.0
        dtype: float64
```

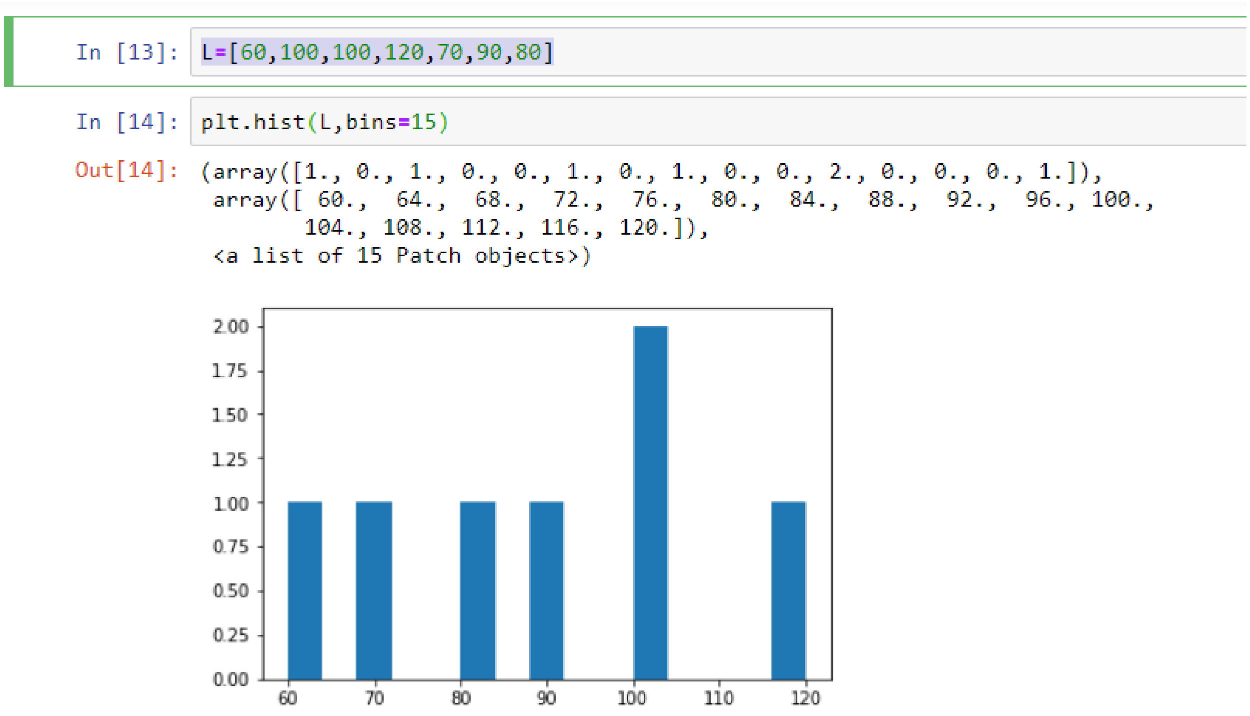
b.



```
In [10]: import matplotlib.pyplot as plt

In [11]: Q=[80,150,135,165,95,130,110]
```

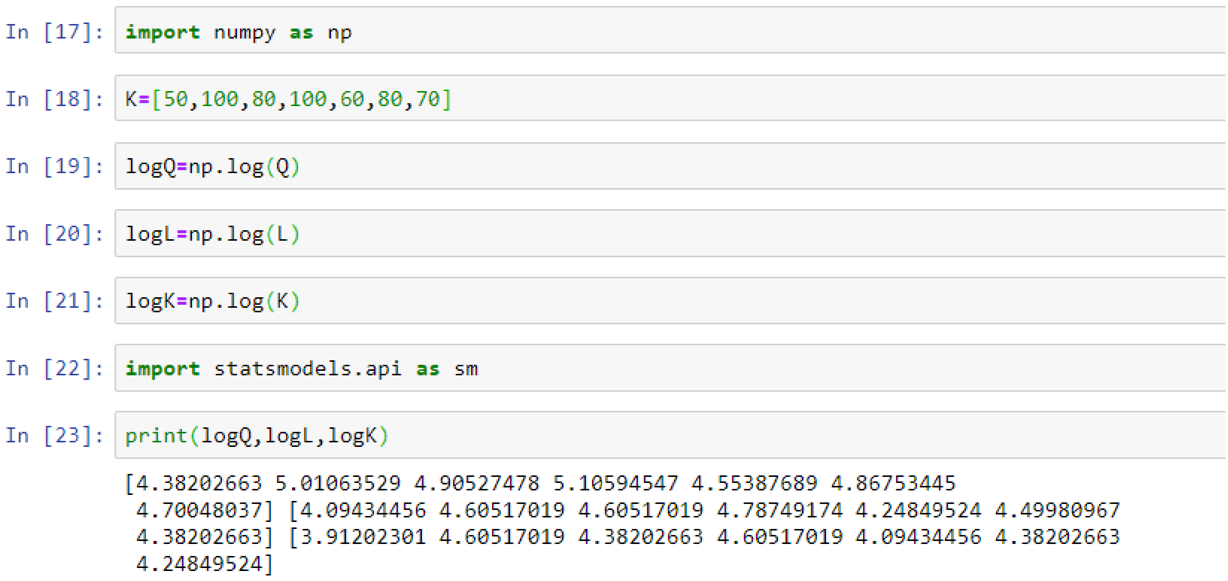


```
In [12]: plt.hist(Q,bins=22)

Out[12]: (array([1., 0., 0., 1., 0., 0., 0., 1., 0., 0., 0., 0., 1., 0., 1., 0., 0.,
                 0., 1., 0., 0., 1.]),
          array([ 80.        ,  83.86363636,  87.72727273,  91.59090909,
                  95.45454545,  99.31818182, 103.18181818, 107.04545455,
                 110.90909091, 114.77272727, 118.63636364, 122.5       ,
                 126.36363636, 130.22727273, 134.09090909, 137.95454545,
                 141.81818182, 145.68181818, 149.54545455, 153.40909091,
                 157.27272727, 161.13636364, 165.        ]),
          <a list of 22 Patch objects>)
```



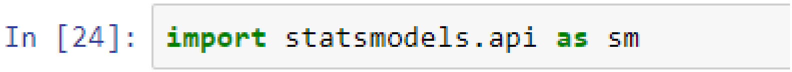
```
In [13]: L=[60,100,100,120,70,90,80]

In [14]: plt.hist(L,bins=15)

Out[14]: (array([1., 0., 1., 0., 0., 1., 0., 1., 0., 0., 2., 0., 0., 0., 1.]),
          array([ 60.,  64.,  68.,  72.,  76.,  80.,  84.,  88.,  92.,  96., 100.,
                 104., 108., 112., 116., 120.]),
          <a list of 15 Patch objects>)
```

C.

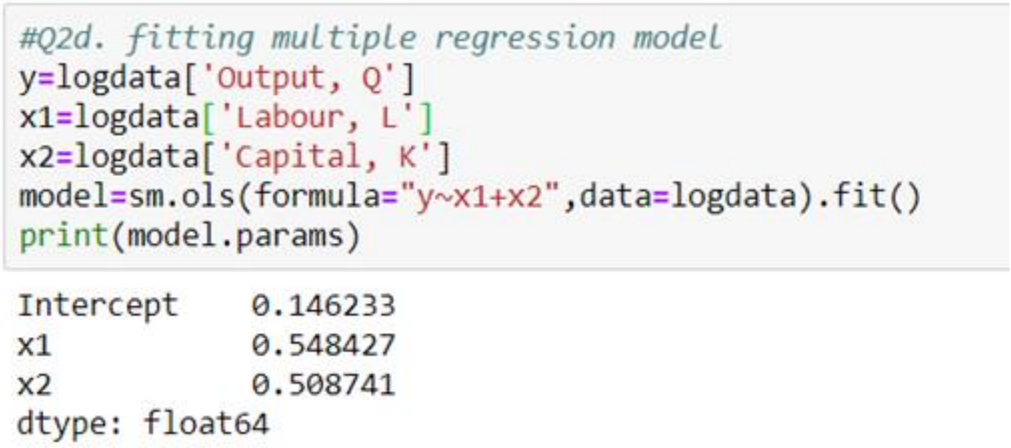


```
In [17]: import numpy as np

In [18]: K=[50,100,80,100,60,80,70]

In [19]: logQ=np.log(Q)

In [20]: logL=np.log(L)

In [21]: logK=np.log(K)

In [22]: import statsmodels.api as sm

In [23]: print(logQ,logL,logK)

         [4.38202663 5.01063529 4.90527478 5.10594547 4.55387689 4.86753445
          4.70048037] [4.09434456 4.60517019 4.60517019 4.78749174 4.24849524 4.49980967
          4.38202663] [3.91202301 4.60517019 4.38202663 4.60517019 4.09434456 4.38202663
          4.24849524]
```

d.

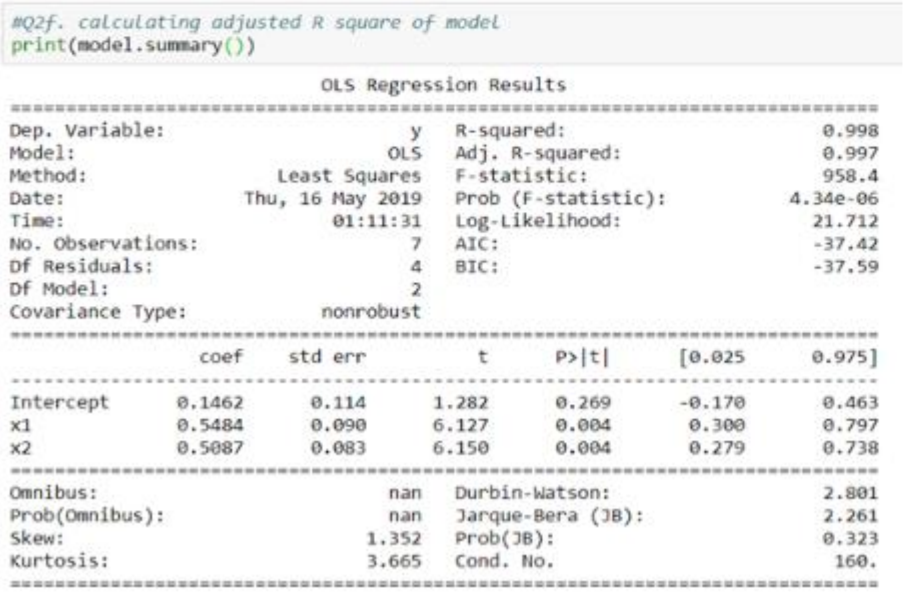


```
In [24]: import statsmodels.api as sm
```



```
#Q2d. fitting multiple regression model
y=logdata['Output, Q']
x1=logdata['Labour, L']
x2=logdata['Capital, K']
model=sm.ols(formula="y~x1+x2",data=logdata).fit()
print(model.params)

Intercept    0.146233
x1           0.548427
x2           0.508741
dtype: float64
```

f.



```
#Q2f. calculating adjusted R square of model
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.998
Model:                            OLS   Adj. R-squared:                  0.997
Method:                 Least Squares   F-statistic:                     958.4
Date:                Thu, 16 May 2019   Prob (F-statistic):           4.34e-06
Time:                        01:11:31   Log-Likelihood:                 21.712
No. Observations:                   7   AIC:                            -37.42
Df Residuals:                       4   BIC:                            -37.59
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.1462      0.114      1.282      0.269      -0.170       0.463
x1             0.5484      0.090      6.127      0.004       0.300       0.797
x2             0.5087      0.083      6.150      0.004       0.279       0.738
==============================================================================
Omnibus:                          nan   Durbin-Watson:                   2.801
Prob(Omnibus):                    nan   Jarque-Bera (JB):                2.261
Skew:                           1.352   Prob(JB):                        0.323
Kurtosis:                       3.665   Cond. No.                         160.
==============================================================================
```