WEEK 1 Python

In [194…
```
#importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import networkx as nx
```

In [74]:
```
#Task 1.1
#importing dataset
dataFrame =  pd.read_csv('Dataset.csv', delimiter=',')
print(dataFrame.to_string())
```

```
    longitude  latitude  housing_median_age  total_rooms  population  median_income  med
ian_house_value ocean_proximity
0     -122.23     37.88             41.0000          880         322         8.3252
452600        NEAR BAY
1     -122.23     37.88             41.0000          880         322         8.3252
452600        NEAR BAY
2     -122.22     37.86             21.0000         7099        2401         8.3014
358500        NEAR BAY
3     -122.25     37.84             52.0001         3104        1157         3.1200
241400        NEAR BAY
4     -122.26     37.85             52.0000         3503        1504         3.2705
241800        NEAR BAY
5     -121.65     39.32             40.0000          812         374         2.7891
73500          INLAND
6     -121.69     39.36             29.0000         2220        1170         2.3224
56200          INLAND
7     -121.70     39.37             32.0000         1852         911         1.7885
57000          INLAND
8     -121.70     39.36             46.0000         1210         523         1.9100
63900          INLAND
9     -121.70     39.36             37.0000         2330        1505         2.0474
56000          INLAND
10    -121.69     39.36             34.0000          842         635         1.8355
63000          INLAND
11    -121.74     39.38             27.0000         2596        1100         2.3243
85500             NaN
12    -121.80     39.33             30.0000         1019         501         2.5259
81300          INLAND
13    -120.46     38.15             16.0000         4221        1516         2.3816
116000          INLAND
14    -120.55     38.12             10.0000         1566         785         2.5000
116100          INLAND
15    -120.56     38.09             34.0000         2745        1150         2.3654
94900          INLAND
16    -124.23     41.75             11.0000         3159        1343         2.4805
73200      NEAR OCEAN
17    -124.21     41.77             17.0000         3461        1947         2.5795
68400          NEAR O
18    -124.19     41.78             15.0000         3140        1645         1.6654
74600          NEAR O
19    -124.16     41.74             15.0000         2715        1532         2.1829
69500      NEAR OCEAN
20    -124.14     41.95             21.0000         2696        1208            NaN
122400      NEAR OCEAN
21    -124.16     41.92             19.0000         1668         841         2.1336
75000      NEAR OCEAN
```

```
22    -118.32     33.35             27.0000          1675          744          2.1579
450000          ISLAND
23    -118.33     33.34             52.0000          2359         1100          2.8333
414700          ISLAND
24    -118.32     33.33             52.0000          2127          733          3.3906
300000          ISLAND
25    -118.32     33.34             52.0000           996          341          2.7361
450000          ISLAND
26    -118.48     33.43             29.0000           716          422          2.6042
287500          ISLAND
27    -118.48     33.43             29.0000           716          422          2.6042
287500          ISLAND
```

In [75]:
```
#Task 1.2 Remove the row with missing values
dataFrame1 = dataFrame.dropna()
# Display the new dataset
print(dataFrame1.to_string())
```

```
    longitude  latitude  housing_median_age  total_rooms  population  median_income  med
ian_house_value ocean_proximity
0     -122.23     37.88             41.0000          880          322          8.3252
452600        NEAR BAY
1     -122.23     37.88             41.0000          880          322          8.3252
452600        NEAR BAY
2     -122.22     37.86             21.0000         7099         2401          8.3014
358500        NEAR BAY
3     -122.25     37.84             52.0001         3104         1157          3.1200
241400        NEAR BAY
4     -122.26     37.85             52.0000         3503         1504          3.2705
241800        NEAR BAY
5     -121.65     39.32             40.0000          812          374          2.7891
73500          INLAND
6     -121.69     39.36             29.0000         2220         1170          2.3224
56200          INLAND
7     -121.70     39.37             32.0000         1852          911          1.7885
57000          INLAND
8     -121.70     39.36             46.0000         1210          523          1.9100
63900          INLAND
9     -121.70     39.36             37.0000         2330         1505          2.0474
56000          INLAND
10    -121.69     39.36             34.0000          842          635          1.8355
63000          INLAND
12    -121.80     39.33             30.0000         1019          501          2.5259
81300          INLAND
13    -120.46     38.15             16.0000         4221         1516          2.3816
116000         INLAND
14    -120.55     38.12             10.0000         1566          785          2.5000
116100         INLAND
15    -120.56     38.09             34.0000         2745         1150          2.3654
94900          INLAND
16    -124.23     41.75             11.0000         3159         1343          2.4805
73200      NEAR OCEAN
17    -124.21     41.77             17.0000         3461         1947          2.5795
68400         NEAR O
18    -124.19     41.78             15.0000         3140         1645          1.6654
74600         NEAR O
19    -124.16     41.74             15.0000         2715         1532          2.1829
69500      NEAR OCEAN
21    -124.16     41.92             19.0000         1668          841          2.1336
75000      NEAR OCEAN
22    -118.32     33.35             27.0000         1675          744          2.1579
450000         ISLAND
23    -118.33     33.34             52.0000         2359         1100          2.8333
414700         ISLAND
24    -118.32     33.33             52.0000         2127          733          3.3906
```

```
300000          ISLAND
25    -118.32     33.34            52.0000          996          341          2.7361
450000          ISLAND
26    -118.48     33.43            29.0000          716          422          2.6042
287500          ISLAND
27    -118.48     33.43            29.0000          716          422          2.6042
287500          ISLAND
```

In [76]:
```
#Task 1.3
#Examining the dataset and resolving the issues
# Examinig Duplicates from dataframe
print(dataFrame1.duplicated())
```

```
0     False
1      True
2     False
3     False
4     False
5     False
6     False
7     False
8     False
9     False
10    False
12    False
13    False
14    False
15    False
16    False
17    False
18    False
19    False
21    False
22    False
23    False
24    False
25    False
26    False
27     True
dtype: bool
```

In [77]:
```
# Remove duplicated rows
dataFrame2 = dataFrame1.drop_duplicates()
dataFrame2.reset_index(drop=True, inplace=True)
# Print data frame
print(dataFrame2.to_string())
```

```
    longitude  latitude  housing_median_age  total_rooms  population  median_income  med
ian_house_value ocean_proximity
0     -122.23     37.88             41.0000          880          322          8.3252
452600        NEAR BAY
1     -122.22     37.86             21.0000         7099         2401          8.3014
358500        NEAR BAY
2     -122.25     37.84             52.0001         3104         1157          3.1200
241400        NEAR BAY
3     -122.26     37.85             52.0000         3503         1504          3.2705
241800        NEAR BAY
4     -121.65     39.32             40.0000          812          374          2.7891
73500          INLAND
5     -121.69     39.36             29.0000         2220         1170          2.3224
56200          INLAND
6     -121.70     39.37             32.0000         1852          911          1.7885
57000          INLAND
7     -121.70     39.36             46.0000         1210          523          1.9100
```

```
63900          INLAND
8     -121.70     39.36            37.0000         2330         1505         2.0474
56000          INLAND
9     -121.69     39.36            34.0000          842          635         1.8355
63000          INLAND
10    -121.80     39.33            30.0000         1019          501         2.5259
81300          INLAND
11    -120.46     38.15            16.0000         4221         1516         2.3816
116000          INLAND
12    -120.55     38.12            10.0000         1566          785         2.5000
116100          INLAND
13    -120.56     38.09            34.0000         2745         1150         2.3654
94900          INLAND
14    -124.23     41.75            11.0000         3159         1343         2.4805
73200      NEAR OCEAN
15    -124.21     41.77            17.0000         3461         1947         2.5795
68400          NEAR O
16    -124.19     41.78            15.0000         3140         1645         1.6654
74600          NEAR O
17    -124.16     41.74            15.0000         2715         1532         2.1829
69500      NEAR OCEAN
18    -124.16     41.92            19.0000         1668          841         2.1336
75000      NEAR OCEAN
19    -118.32     33.35            27.0000         1675          744         2.1579
450000          ISLAND
20    -118.33     33.34            52.0000         2359         1100         2.8333
414700          ISLAND
21    -118.32     33.33            52.0000         2127          733         3.3906
300000          ISLAND
22    -118.32     33.34            52.0000          996          341         2.7361
450000          ISLAND
23    -118.48     33.43            29.0000          716          422         2.6042
287500          ISLAND
```

In [78]:
```
#Task 1.4 Wrong data types
# correcting Wrong data types
dataFrame2.info()
dataFrame2.at[3, 'housing_median_age'] = int(dataFrame2.at[3, 'housing_median_age'])
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24 entries, 0 to 23
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           24 non-null     float64
 1   latitude            24 non-null     float64
 2   housing_median_age  24 non-null     float64
 3   total_rooms         24 non-null     int64
 4   population          24 non-null     int64
 5   median_income       24 non-null     float64
 6   median_house_value  24 non-null     int64
 7   ocean_proximity     24 non-null     object
dtypes: float64(4), int64(3), object(1)
memory usage: 1.6+ KB
```

In [80]:
```
#Task 1.5 Correcting the Wrong values
dataFrame2.at[15, 'ocean_proximity'] = 'NEAR OCEAN'
dataFrame2.at[16, 'ocean_proximity'] = 'NEAR OCEAN'
print(dataFrame2.to_string())
```

```
    longitude  latitude  housing_median_age  total_rooms  population  median_income  med
ian_house_value ocean_proximity
0     -122.23     37.88             41.0000          880         322         8.3252
452600        NEAR BAY
1     -122.22     37.86             21.0000         7099        2401         8.3014
```

```
358500        NEAR BAY
2     -122.25     37.84            52.0001         3104         1157         3.1200
241400        NEAR BAY
3     -122.26     37.85            52.0000         3503         1504         3.2705
241800        NEAR BAY
4     -121.65     39.32            40.0000          812          374         2.7891
73500          INLAND
5     -121.69     39.36            29.0000         2220         1170         2.3224
56200          INLAND
6     -121.70     39.37            32.0000         1852          911         1.7885
57000          INLAND
7     -121.70     39.36            46.0000         1210          523         1.9100
63900          INLAND
8     -121.70     39.36            37.0000         2330         1505         2.0474
56000          INLAND
9     -121.69     39.36            34.0000          842          635         1.8355
63000          INLAND
10    -121.80     39.33            30.0000         1019          501         2.5259
81300          INLAND
11    -120.46     38.15            16.0000         4221         1516         2.3816
116000         INLAND
12    -120.55     38.12            10.0000         1566          785         2.5000
116100         INLAND
13    -120.56     38.09            34.0000         2745         1150         2.3654
94900          INLAND
14    -124.23     41.75            11.0000         3159         1343         2.4805
73200      NEAR OCEAN
15    -124.21     41.77            17.0000         3461         1947         2.5795
68400      NEAR OCEAN
16    -124.19     41.78            15.0000         3140         1645         1.6654
74600      NEAR OCEAN
17    -124.16     41.74            15.0000         2715         1532         2.1829
69500      NEAR OCEAN
18    -124.16     41.92            19.0000         1668          841         2.1336
75000      NEAR OCEAN
19    -118.32     33.35            27.0000         1675          744         2.1579
450000         ISLAND
20    -118.33     33.34            52.0000         2359         1100         2.8333
414700         ISLAND
21    -118.32     33.33            52.0000         2127          733         3.3906
300000         ISLAND
22    -118.32     33.34            52.0000          996          341         2.7361
450000         ISLAND
23    -118.48     33.43            29.0000          716          422         2.6042
287500         ISLAND
```

In [81]:
```
#Task 1.6
#saving the data into new file
uData = dataFrame2.copy()
uData.to_csv('uDataset.csv', index=False)
```

In [82]:
```
#Task 1.7
#Mean of the Updated Data
uData.median_house_value.mean()
```

Out[82]: 180629.16666666666

In [83]:
```
#Task 1.8
#Median of the Updated Data
uData.median_house_value.median()
```

Out[83]: 88100.0

In [84]:
```
#Task 1.9
#Range
print("Max Range: ",uData.median_house_value.max())
print("Min Range: ",uData.median_house_value.min())
```

```
Max Range:  452600
Min Range:  56000
```

In [85]:
```
#Task 1.10
#Range
uData.median_income = uData.median_income*10000
print(uData.to_string())
```

```
    longitude  latitude  housing_median_age  total_rooms  population  median_income  med
ian_house_value ocean_proximity
0     -122.23     37.88             41.0000          880         322        83252.0
452600        NEAR BAY
1     -122.22     37.86             21.0000         7099        2401        83014.0
358500        NEAR BAY
2     -122.25     37.84             52.0001         3104        1157        31200.0
241400        NEAR BAY
3     -122.26     37.85             52.0000         3503        1504        32705.0
241800        NEAR BAY
4     -121.65     39.32             40.0000          812         374        27891.0
73500          INLAND
5     -121.69     39.36             29.0000         2220        1170        23224.0
56200          INLAND
6     -121.70     39.37             32.0000         1852         911        17885.0
57000          INLAND
7     -121.70     39.36             46.0000         1210         523        19100.0
63900          INLAND
8     -121.70     39.36             37.0000         2330        1505        20474.0
56000          INLAND
9     -121.69     39.36             34.0000          842         635        18355.0
63000          INLAND
10    -121.80     39.33             30.0000         1019         501        25259.0
81300          INLAND
11    -120.46     38.15             16.0000         4221        1516        23816.0
116000          INLAND
12    -120.55     38.12             10.0000         1566         785        25000.0
116100          INLAND
13    -120.56     38.09             34.0000         2745        1150        23654.0
94900          INLAND
14    -124.23     41.75             11.0000         3159        1343        24805.0
73200      NEAR OCEAN
15    -124.21     41.77             17.0000         3461        1947        25795.0
68400      NEAR OCEAN
16    -124.19     41.78             15.0000         3140        1645        16654.0
74600      NEAR OCEAN
17    -124.16     41.74             15.0000         2715        1532        21829.0
69500      NEAR OCEAN
18    -124.16     41.92             19.0000         1668         841        21336.0
75000      NEAR OCEAN
19    -118.32     33.35             27.0000         1675         744        21579.0
450000          ISLAND
20    -118.33     33.34             52.0000         2359        1100        28333.0
414700          ISLAND
21    -118.32     33.33             52.0000         2127         733        33906.0
300000          ISLAND
22    -118.32     33.34             52.0000          996         341        27361.0
450000          ISLAND
23    -118.48     33.43             29.0000          716         422        26042.0
287500          ISLAND
```

WEEK 2

In [168…
```
#part 1

#uniform distributed array data
array=np.empty(shape=(5,10), dtype='int')
for i in range(0,5):
    for j in range(0,10):
        array[i][j] = int(np.random.randint(0,9))

array
```

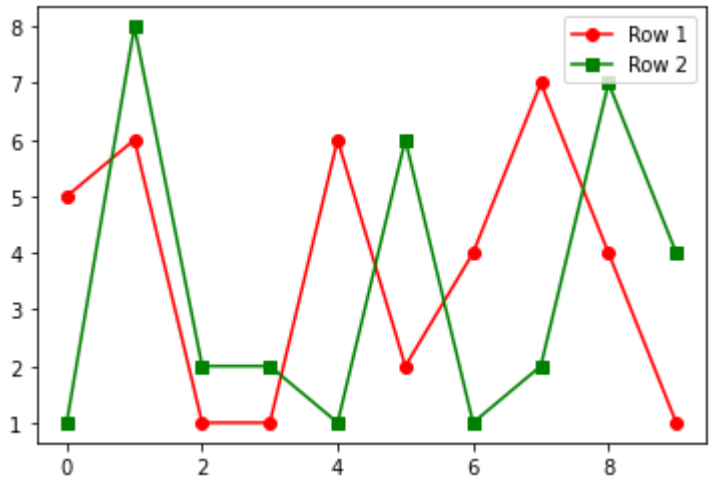
Out[168…
```
array([[5, 6, 1, 1, 6, 2, 4, 7, 4, 1],
       [1, 8, 2, 2, 1, 6, 1, 2, 7, 4],
       [7, 3, 7, 2, 2, 7, 4, 6, 0, 1],
       [0, 7, 2, 0, 4, 4, 8, 8, 6, 7],
       [5, 0, 6, 6, 3, 1, 1, 0, 6, 4]])
```

In [169…
```
# Converting two dimensional array to table format
df = pd.DataFrame(array, columns = ['0','1','2','3','4','5','6','7','8','9'])
df
```

Out[169…

	0	1	2	3	4	5	6	7	8	9
0	5	6	1	1	6	2	4	7	4	1
1	1	8	2	2	1	6	1	2	7	4
2	7	3	7	2	2	7	4	6	0	1
3	0	7	2	0	4	4	8	8	6	7
4	5	0	6	6	3	1	1	0	6	4

In [172…
```
#ploting and showing the result of first 2 rows
plt.plot(array[0,],marker='o', label = 'Row 1', color="r")
plt.plot(array[1,], marker='s', label = 'Row 2', color="g")
plt.legend()
plt.show()
```



In [137…
```
## part2
#part2.1
```

```
#Mean
df.mean()
```

Out[137…
```
0    4.4
1    4.0
2    5.8
3    4.6
4    5.2
5    4.8
6    4.0
7    4.6
8    5.4
9    2.6
dtype: float64
```

In [138…
```
#part2.2

#Median
df.median()
```

Out[138…
```
0    5.0
1    4.0
2    7.0
3    5.0
4    6.0
5    6.0
6    4.0
7    4.0
8    6.0
9    2.0
dtype: float64
```

In [139…
```
#part2.3

#Stander Deviation
df.std()
```

Out[139…
```
0    2.701851
1    1.870829
2    2.949576
3    3.049590
4    2.588436
5    2.949576
6    2.915476
7    2.190890
8    2.302173
9    2.792848
dtype: float64
```

In [140…
```
## part 3

ar1 = np.random.normal(loc=17, scale=0.2, size=1000)
ar1
```

Out[140…
```
array([17.19385308, 17.11781519, 16.9419913 , 16.94469713, 17.08165089,
       17.01651947, 16.9983142 , 17.07621753, 17.22451131, 17.09102663,
       17.03871232, 16.49609872, 16.92296862, 17.18471015, 17.27544419,
       17.15219315, 17.02874491, 17.15520832, 16.65210474, 17.01053232,
       16.81223307, 16.75961737, 17.16039944, 17.00870432, 17.03459292,
       16.87449953, 17.00391494, 16.8396176 , 16.93910114, 16.9996121 ,
       16.88420875, 16.78778003, 17.2188295 , 16.96836919, 16.81408588,
       17.13843904, 16.92644861, 16.90787621, 17.05136472, 16.80247516,
       17.17773425, 17.33832267, 17.21114944, 17.06351754, 17.51160834,
```

```
17.16137473, 16.68445951, 16.95518926, 16.74432088, 17.33398257,
16.7512898 , 16.85638191, 17.01893366, 17.26469343, 16.88736048,
17.14498765, 16.97348878, 16.74594199, 16.70446213, 16.97514822,
16.90669148, 16.85057521, 17.11215798, 16.8201721 , 17.11728241,
17.17651342, 16.79246237, 17.18038791, 17.2968802 , 17.17176642,
16.90330059, 17.27639423, 17.06559102, 16.99618205, 17.00181192,
17.11192493, 17.28051535, 16.94639561, 17.07225263, 17.1572567 ,
16.75669556, 17.05247704, 17.13019201, 16.85538206, 16.84104529,
16.98568566, 16.94145646, 16.67433805, 17.60695269, 17.11168773,
16.86816116, 17.17854282, 17.08357196, 17.19801307, 17.13880665,
17.11142414, 16.74609229, 16.83119048, 17.09403786, 16.96392817,
16.47670876, 17.17540243, 16.67674477, 17.12197671, 16.88891672,
16.7610628 , 16.80017956, 17.10651085, 16.72902923, 17.31874912,
16.80911327, 16.8786832 , 17.30167381, 17.02490295, 16.74178732,
16.99061329, 17.0444523 , 16.75197758, 17.121929  , 16.92154509,
17.56132015, 17.32755808, 17.00343217, 17.24427117, 17.09086451,
16.79028341, 17.14106214, 17.30745469, 16.90974102, 16.95399942,
17.11588542, 17.11566031, 17.21995777, 17.01687429, 17.12883065,
16.67967115, 17.10330594, 16.9892246 , 17.14992751, 16.9317303 ,
17.18444649, 16.94826798, 17.05144592, 17.03272946, 17.30547593,
16.76769295, 16.86635108, 16.94144789, 17.38611971, 17.34745177,
17.16799   , 17.51142678, 17.06360879, 16.86961638, 16.92495447,
16.96126941, 16.6624198 , 16.51305518, 16.78092689, 16.93938572,
17.10994486, 17.04905825, 17.07003151, 16.95219004, 17.04103899,
16.89784759, 17.11363126, 17.04937404, 17.15538562, 17.27777935,
17.14459616, 16.80961774, 17.11425386, 17.28684425, 16.97191309,
17.05635681, 16.75740119, 16.92669366, 16.88885764, 16.9214244 ,
16.81142644, 16.63496854, 16.84212033, 17.21174994, 16.94238334,
17.16638694, 16.9724666 , 16.72879296, 17.27299601, 17.22337056,
17.03754792, 16.91493726, 17.0573827 , 16.99616058, 17.17508728,
16.95471919, 17.29126056, 17.20215883, 16.98901616, 17.05460216,
16.8038945 , 16.88458089, 17.19386285, 17.47919501, 16.99788309,
17.13750849, 17.03917412, 16.61507531, 16.94565164, 16.68230669,
16.72309434, 16.70636031, 16.98989252, 16.81135829, 17.01801019,
17.2855583 , 16.97835846, 17.15819873, 16.97946593, 16.86124642,
16.72708241, 17.19199276, 17.33326553, 17.0502102 , 17.2321407 ,
17.05240431, 16.89334126, 17.14617095, 17.06194204, 17.27880689,
16.84429223, 17.10821885, 16.77694589, 16.80743774, 16.94313489,
16.96728416, 17.19844313, 17.16923452, 17.13086396, 17.0053554 ,
16.74284393, 17.15947983, 16.79776857, 17.16558937, 16.99951373,
16.96835812, 17.26949071, 16.9539783 , 16.96954324, 17.19454648,
17.11997603, 16.86543129, 16.7905238 , 17.11034784, 16.96483464,
17.17112937, 17.57913088, 17.21166687, 17.06025868, 16.91785566,
17.10367549, 16.9586354 , 17.0141769 , 17.34635543, 17.1468895 ,
17.29679252, 17.06826676, 16.87099486, 16.96812228, 17.33943953,
16.9059965 , 16.91112033, 16.88387895, 16.85305543, 17.0522515 ,
16.84244414, 16.85139199, 17.042532  , 17.13362794, 17.25408578,
17.4231503 , 16.95027357, 16.79162975, 16.74985047, 17.04887886,
17.26885051, 17.02377346, 16.75412099, 16.77023085, 17.11933612,
17.1919573 , 16.93367268, 17.16207063, 16.88106843, 16.7329988 ,
17.35025567, 17.25399153, 16.97385405, 17.24868069, 16.61362281,
17.14270388, 17.0308246 , 17.19619563, 17.32003691, 17.04017343,
16.85626128, 17.34648586, 17.29839536, 17.3020593 , 17.1228724 ,
17.05722503, 17.15795585, 16.86136048, 16.83098041, 16.68180492,
17.19447566, 17.15221289, 16.96018353, 16.9058855 , 17.10568147,
17.22446854, 17.26515872, 17.4544066 , 17.08437084, 17.39674114,
17.31215446, 17.12562139, 17.10173167, 17.02390823, 16.85431703,
16.98864713, 17.21305644, 17.00019704, 16.79563827, 16.95679566,
17.01962116, 17.08491448, 17.13171469, 17.22491383, 16.30480393,
17.3583236 , 17.20779648, 17.05553557, 17.1468314 , 17.236134  ,
16.90518489, 17.3070877 , 17.0534648 , 16.56079711, 17.13268308,
17.34842521, 16.99799712, 16.88281379, 16.85553526, 17.18991058,
17.13863094, 17.11203354, 17.15430972, 17.33637112, 17.48724957,
17.17733466, 16.63775377, 17.12114836, 16.979646  , 16.81831558,
16.94399271, 17.11815028, 17.12997644, 17.23851074, 17.19085412,
```

```
17.14044254, 16.92128482, 16.78672943, 16.78090133, 16.5845147 ,
17.04938121, 16.79047518, 17.11207294, 17.11770524, 17.00925455,
16.96242871, 17.14262694, 16.9021022 , 16.8808276 , 16.7285508 ,
17.03969507, 16.91830462, 17.05721588, 17.1313047 , 16.73447852,
17.51371525, 17.16976828, 16.98095654, 17.27256745, 17.02298101,
17.43114164, 17.15744558, 17.12263544, 17.10262939, 16.94611753,
16.79556521, 16.87393837, 17.13637673, 17.14220936, 17.06339356,
16.96680918, 16.83005431, 16.96987181, 16.98094357, 16.89154479,
16.70443242, 16.94749683, 17.02503837, 17.16085328, 16.7600398 ,
16.89555925, 17.14164232, 16.76994525, 16.71763239, 16.99836769,
17.23622279, 16.8371685 , 16.97952547, 16.81489303, 16.95959802,
17.2242386 , 16.88157136, 17.12713567, 16.9370056 , 17.18100742,
17.21786495, 16.74745477, 17.13572991, 17.10015875, 16.89477858,
16.74184984, 16.62898962, 17.08198691, 17.17942144, 16.71541154,
17.01760585, 17.08680275, 16.90711239, 17.05300666, 16.5841985 ,
17.01051669, 16.76737917, 17.21979256, 16.86377711, 17.25413694,
17.00059869, 16.66450704, 16.90089321, 17.18544566, 17.10706589,
16.98900015, 17.11093186, 17.04542171, 16.58945686, 16.8008222 ,
17.16344234, 16.77547028, 16.9935044 , 16.91695278, 17.16220307,
16.74291431, 16.5811604 , 17.05936616, 16.90675372, 16.87863612,
17.1597134 , 16.81581282, 16.85697232, 17.23246264, 16.90492939,
17.13171426, 16.89813315, 17.08893872, 17.22568894, 16.98612676,
17.17698706, 16.74325915, 17.12791045, 17.01494773, 17.036365  ,
16.65972743, 17.01060014, 16.96585   , 16.79520636, 17.13671319,
16.89093665, 17.11298094, 17.47783632, 17.119607  , 16.89982118,
16.95347098, 17.0095219 , 16.95560106, 16.80500475, 16.86463455,
16.63772386, 17.07521493, 16.96047566, 16.66953679, 16.98871491,
16.56391635, 17.28989149, 16.90077422, 16.99671875, 17.26944083,
16.49634195, 16.83883435, 17.01803838, 16.72660206, 16.93822312,
16.98507694, 17.03256535, 17.12402575, 17.21738863, 17.17292461,
16.68898747, 16.97121652, 17.0765408 , 16.66529301, 16.56886997,
16.9545326 , 17.30678601, 17.09230989, 17.25760488, 16.77621278,
17.09844273, 17.21303496, 16.82372109, 16.93534771, 17.19377202,
17.12969186, 16.72677882, 16.88765006, 16.6427106 , 17.041662  ,
16.6176479 , 16.90282665, 17.40013356, 17.06255626, 16.64500226,
16.86304371, 17.18561229, 17.18394934, 16.91840903, 16.55646657,
16.66902137, 17.29394369, 17.13697514, 16.94886574, 16.95956565,
16.80432631, 16.57272233, 17.09673221, 17.21057259, 16.99953565,
17.21909726, 16.98941698, 16.81708945, 17.01781984, 16.80284637,
17.30478669, 17.11334515, 17.27755623, 16.85240704, 17.25097984,
17.17628617, 17.51301392, 16.68538846, 16.85685778, 17.07273002,
16.90749266, 17.12938929, 16.89005889, 16.64593383, 17.42985443,
16.57197657, 17.0255929 , 16.82215297, 16.81685227, 16.76684092,
17.37372599, 17.22479406, 16.93215954, 17.60024858, 16.83920111,
16.90392976, 17.31860099, 16.84852922, 16.99474626, 17.19367225,
17.07488458, 17.3674609 , 16.88420072, 16.62710627, 16.97883135,
16.8309237 , 16.91215996, 17.05130653, 16.82342521, 17.17358769,
17.20252627, 16.90841221, 17.04404584, 16.93216778, 16.73521857,
16.92119261, 16.95100428, 17.05168915, 17.2287935 , 17.37968013,
17.11913699, 17.40400272, 16.93092235, 17.02016379, 16.82048118,
16.96959809, 16.77166456, 17.30866698, 17.14372894, 17.19899822,
17.17623373, 17.00004514, 17.01150987, 16.86357675, 17.15309773,
16.68677594, 17.26872621, 17.38396654, 17.01092102, 16.90834304,
16.62538039, 17.11674463, 17.16152495, 17.21284927, 16.67156277,
17.19287879, 16.9870709 , 17.08283318, 17.29538416, 16.84343316,
17.03703267, 17.3740284 , 17.05420479, 17.24809381, 16.85399395,
17.035122  , 16.71936049, 16.98189906, 17.01428217, 17.27260072,
16.90491358, 17.18481642, 17.26609656, 17.23325667, 17.13251736,
17.02789712, 16.75887254, 16.92144841, 16.95076066, 16.82946309,
17.07393242, 17.52899818, 17.19227218, 16.98699768, 16.99474731,
16.83526816, 16.7431376 , 17.35137348, 16.81639189, 17.10627031,
17.03633712, 16.66035583, 17.11130502, 17.12176337, 17.02212315,
16.74102113, 16.54324557, 17.0511258 , 17.10697101, 16.6509754 ,
16.71338504, 16.7622198 , 16.90531426, 17.07054964, 16.54242777,
16.81808707, 16.81108807, 16.96914761, 17.19197383, 17.14793576,
```

```
17.07497799, 17.43309661, 17.03476291, 16.87388834, 17.04792986,
17.25063962, 16.93851726, 16.88954348, 16.86860122, 16.60542342,
16.39305872, 17.01240678, 16.94703898, 17.52966443, 17.3361454 ,
16.99703281, 17.40463058, 16.78430871, 16.94899645, 16.96689974,
16.99741422, 16.97741917, 16.95711894, 16.91451096, 17.09618073,
17.02446197, 16.62845221, 17.08297523, 17.04611179, 16.85557743,
17.0398967 , 16.92566916, 16.721329  , 17.31338665, 16.80753604,
17.11382619, 17.06922728, 16.98081176, 16.77430087, 17.08376414,
16.98329457, 17.00260124, 17.32504998, 16.97522858, 17.0951949 ,
16.95250956, 16.8760521 , 17.17006801, 16.72310058, 16.87169899,
16.80053746, 17.18553501, 16.89387085, 17.18181117, 16.89764792,
16.82281243, 17.14884545, 17.24413004, 16.86806673, 17.05312707,
16.97775711, 17.18358847, 16.86440377, 16.672454  , 17.3389735 ,
16.9538856 , 16.84189806, 16.77697438, 16.67871932, 16.80968517,
16.79233828, 16.93916561, 17.10104626, 17.08972512, 17.11421849,
17.1159912 , 17.15582178, 16.78345059, 16.98931051, 16.92884145,
17.21004254, 16.90021252, 16.89469819, 16.59402025, 16.68939877,
17.07024691, 17.3315622 , 17.07898748, 17.01328798, 17.30030151,
16.90620082, 17.53128155, 17.18384507, 16.71596613, 17.10880518,
17.07733925, 17.17284879, 17.23443967, 17.07796108, 16.93851864,
16.9376072 , 16.943394  , 16.84361327, 17.05445681, 16.92606493,
16.88467941, 17.49164006, 16.98142265, 16.85825284, 17.14377975,
16.83853637, 17.10922821, 16.74719612, 17.34017753, 17.19324629,
16.97007027, 17.41942161, 17.16427494, 17.10674722, 17.17537627,
17.21246227, 16.65869982, 17.20429768, 17.00428172, 16.86074318,
17.05306952, 16.63689931, 17.00376189, 16.95579631, 16.7997598 ,
17.31104845, 16.46839381, 17.09230934, 16.55383938, 16.80742348,
17.09918542, 16.92878143, 16.67798522, 16.71495853, 17.05279835,
17.09695212, 17.2120693 , 17.42738186, 17.07419086, 16.79695344,
16.91544959, 17.08699263, 16.99315709, 16.7844845 , 16.9700712 ,
17.0679022 , 17.11550256, 17.03633846, 16.86861482, 16.98333334,
16.98966082, 16.91434403, 16.67537405, 16.87784758, 16.92078414,
16.6198439 , 16.90165915, 17.1143573 , 17.08476432, 16.95411275,
17.17686812, 16.94923552, 17.15438787, 17.24118882, 16.362624  ,
17.15125419, 16.79000954, 17.05252383, 16.87881156, 17.22651631,
17.0195417 , 16.8298494 , 16.91826809, 17.18942933, 17.06012677,
16.91751863, 16.94546634, 16.52329528, 17.14136019, 17.07533326,
16.73144626, 16.96385521, 17.0504288 , 16.94839328, 16.8524748 ,
17.18184205, 17.28938122, 16.71189637, 17.14698868, 16.47063619,
17.14591812, 16.98946983, 17.09180922, 17.01534923, 17.11691519,
17.5576903 , 17.03452861, 16.81513507, 16.98421663, 17.0046553 ,
17.17943154, 16.84904707, 16.96349844, 16.77667383, 16.85828163,
17.36319728, 16.82774411, 17.05431988, 16.65127652, 16.90998164,
16.89780975, 16.90898239, 17.08611144, 16.85341356, 17.18918008,
16.86327007, 17.11680939, 17.11458756, 17.01877505, 16.95763889,
17.11306445, 16.93443029, 16.9883374 , 16.91899431, 17.1685946 ,
16.98586235, 17.03846099, 17.08350663, 17.12476326, 16.89660624,
17.03360101, 16.79397068, 16.99077899, 16.66953317, 17.03819285,
16.98758137, 17.14675872, 16.86660474, 16.88475639, 17.11966787,
17.08621357, 17.39450022, 16.98271827, 16.63706647, 16.83840186,
17.08568807, 17.22912183, 17.03460905, 16.99861106, 16.83199444,
17.22634944, 17.48492309, 17.23447913, 17.54792157, 17.20095019,
17.18736973, 17.23522573, 16.97325712, 17.06551965, 17.13651451,
16.76631411, 16.95379812, 17.2027339 , 17.14346174, 17.06765629,
17.18945873, 16.7843909 , 16.73514063, 17.09494961, 16.92441435,
17.3279313 , 17.1086076 , 17.30007924, 17.13320142, 16.80650125,
17.14278238, 17.15090404, 17.09965532, 16.7993911 , 16.91713851,
16.65812083, 16.86636664, 16.91921334, 16.87384645, 16.90218803,
16.93224326, 16.93179546, 16.96028051, 16.87531109, 17.06115345,
17.23077739, 16.79491312, 16.76926943, 16.86036165, 16.94603464,
16.88751434, 16.9291848 , 16.93629009, 16.88918628, 17.1345501 ])
```

In [173…

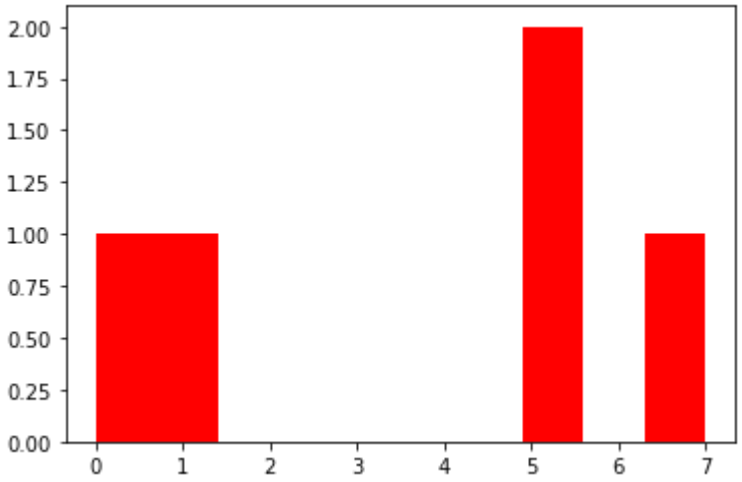
```
## Part 4

# Minimum Value
```

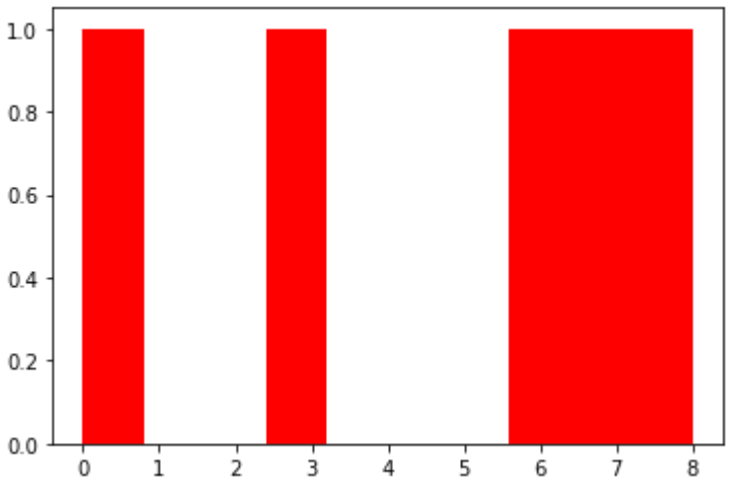
```
print("min :", min(ar1))
# Maximum Value
print("max :", max(ar1))
# Range
rng  = max(ar1) - min(ar1)
print("Range: ",rng)
```

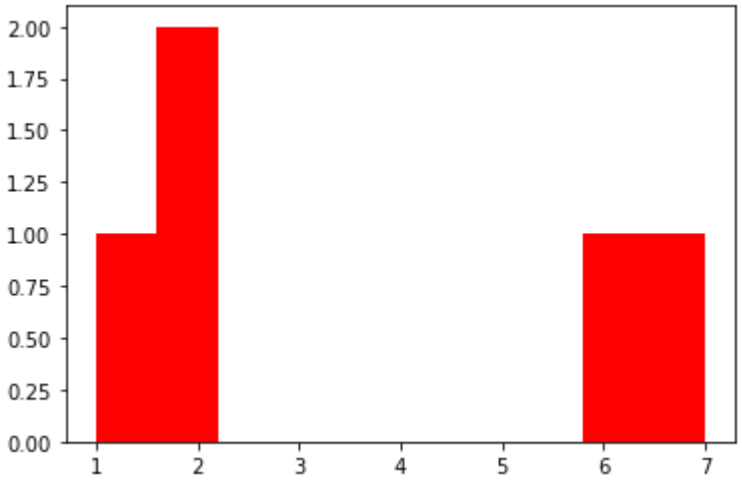
```
min : 16.30480393118943
max : 17.606952688077403
Range:  1.3021487568879735
```

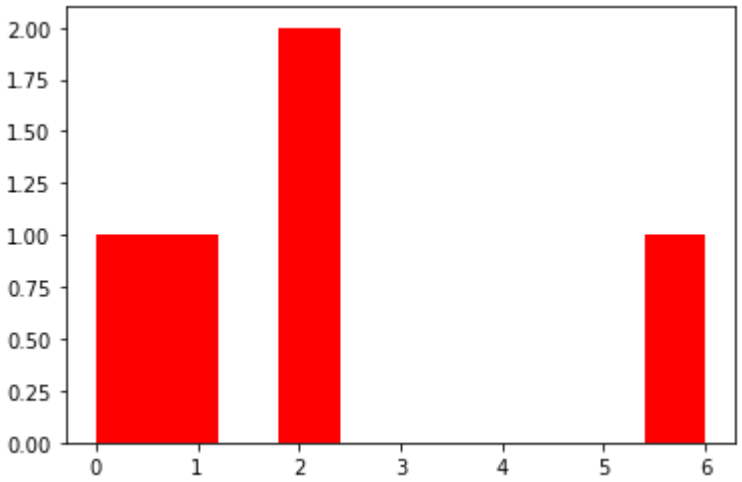
In [181…

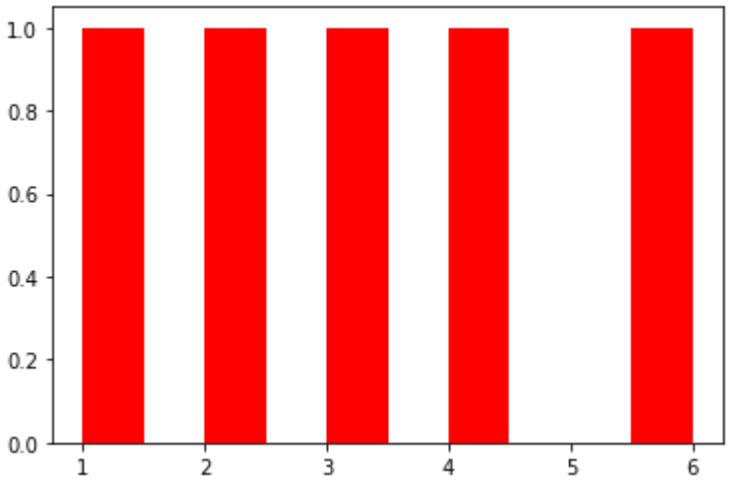
```
## Part 5

# Visualise the dataset by using a histogram with 10 bins
for i, col in enumerate(df.columns):
    plt.figure(i)
    plt.hist(df[col], bins = 10, color="r")
```

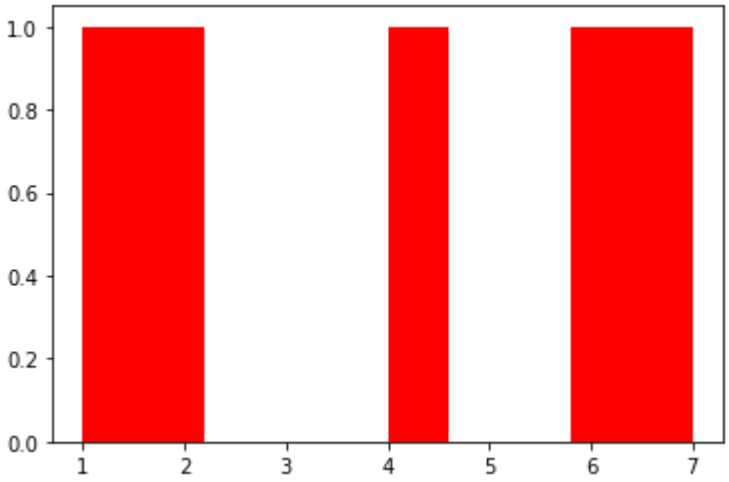


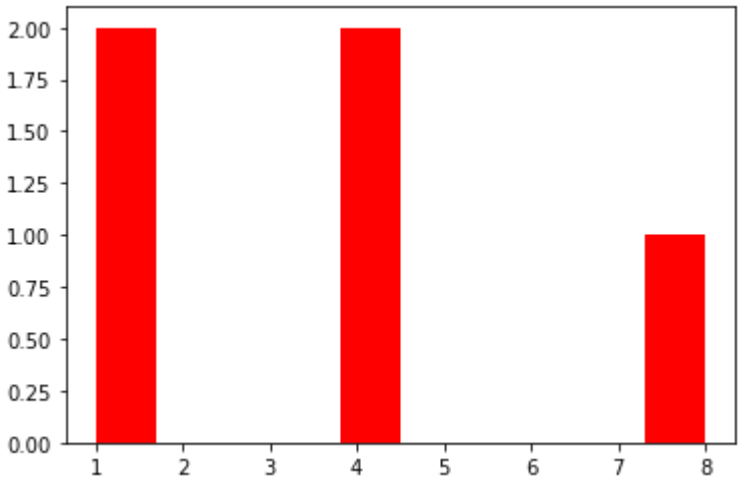


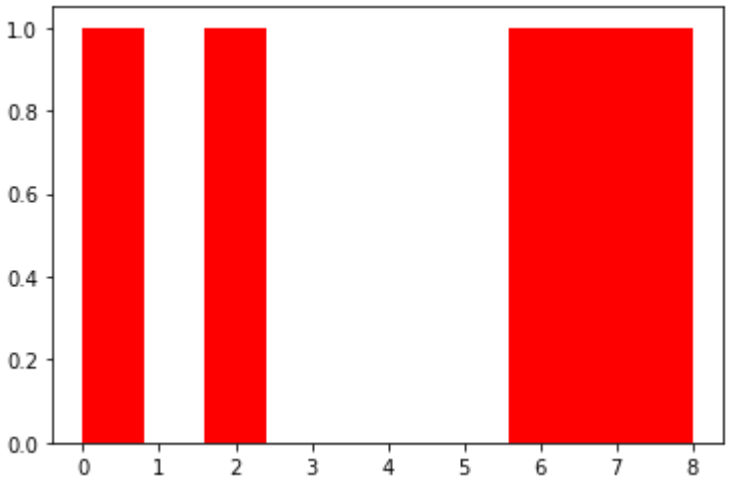

2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
1
2
3
4
5
6
7

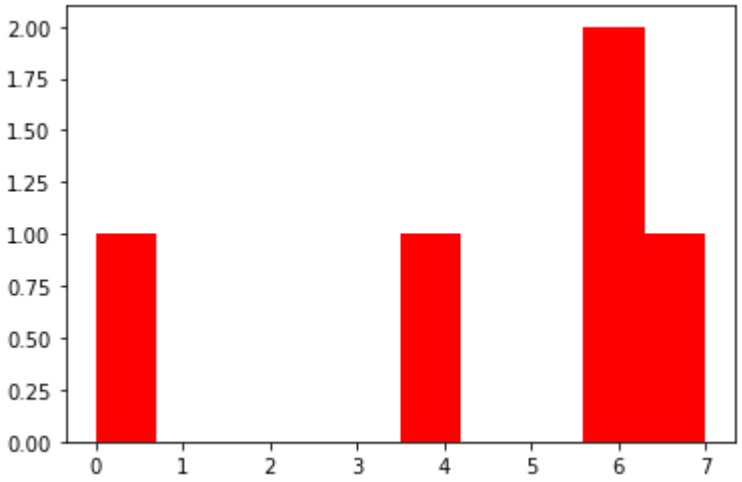

2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
1
2
3
4
5
6

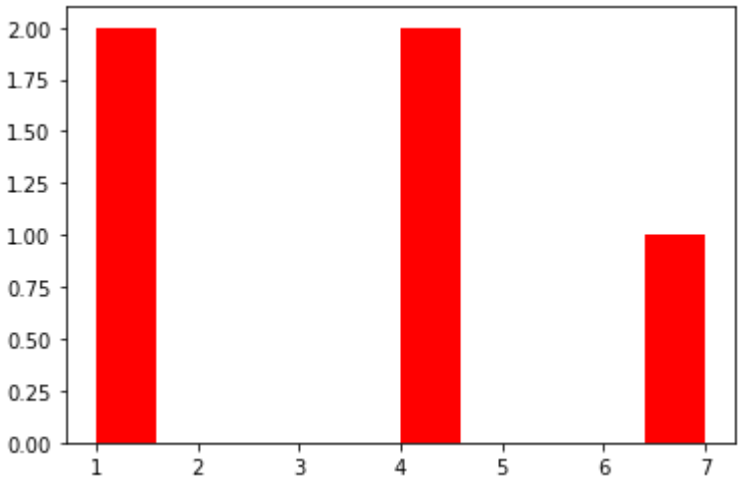

1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6


1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
7


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
1
2
3
4
5
6
7
8

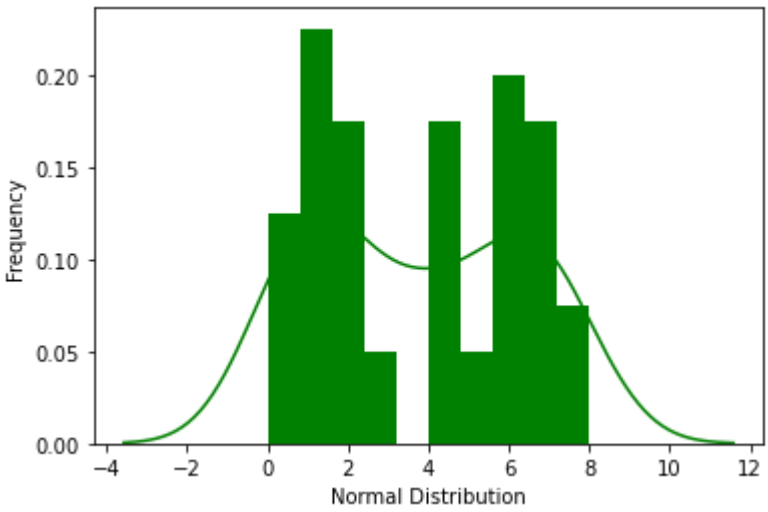

1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
7
8





In [180…

```
# Visualise the probability density function
ax = sns.distplot(df,
                  bins=10,
                  kde=True,
                  color='green',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Normal Distribution', ylabel='Frequency');
```



Week 3

In [183…

```
## Part 1

#initializing the matrix
M = np.array([[2, 5, 1], [4, 3, 7], [1, 3, 2]])
print('A:', M)

#finding Determinant of the matrix
D = np.linalg.det(M)
print('D:', D)

#finding Trace of the matrix
T = np.trace(M)
print('T:', T)

#finding inverse of the matrix
A1 = np.linalg.inv(M)
print('inverse A1:', A1)
```

```
A: [[2 5 1]
 [4 3 7]
 [1 3 2]]
D: -26.000000000000014
T: 7
inverse A1: [[ 0.57692308  0.26923077 -1.23076923]
 [ 0.03846154 -0.11538462  0.38461538]
 [-0.34615385  0.03846154  0.53846154]]
```

In [185…

```
## part 2
#defining the matrix B and C
B = np.array([[4, 7, 2], [3, 2, 5], [6, 4, 3]])
print('B:', B)
C = np.array([[3, 1, 9], [5, 7, 8], [2, 1, 1]])
print('C:', C)


P = np.matmul(B, C)
print('P:', S)
```

```
B: [[4 7 2]
 [3 2 5]
 [6 4 3]]
C: [[3 1 9]
 [5 7 8]
 [2 1 1]]
P: [[51 55 94]
 [29 22 48]
 [44 37 89]]
```

part 3

AB=C

$$A\begin{pmatrix}3 & 2 & -1\\ 2 & -1 & 4\\ 4 & -2 & 3\end{pmatrix}B\begin{pmatrix}X\\ Y\\ Z\end{pmatrix}=C\begin{pmatrix}25\\ 19\\ 18\end{pmatrix} \quad (1)$$

part 4

A'AB=A'C

$$A' \begin{pmatrix} 0.14285714 & -0.11428571 & 0.2 \\ 0.28571429 & 0.37142857 & -0.4 \\ 0. & 0.4 & -0.2 \end{pmatrix} A \begin{pmatrix} 3 & 2 & -1 \\ 2 & -1 & 4 \\ 4 & -2 & 3 \end{pmatrix} B \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = A' \begin{pmatrix} 0.14285714 \\ 0.28571429 \\ 0. \end{pmatrix}$$

In [193…

```
## part 5
A = np.array([[3, 2, -1], [2, -1, 4], [4, -2, 3]])
A1 = np.linalg.inv(A)
C = np.array([[25], [19], [18]])
M = np.matmul(A1, C)
print('M:', X)
```

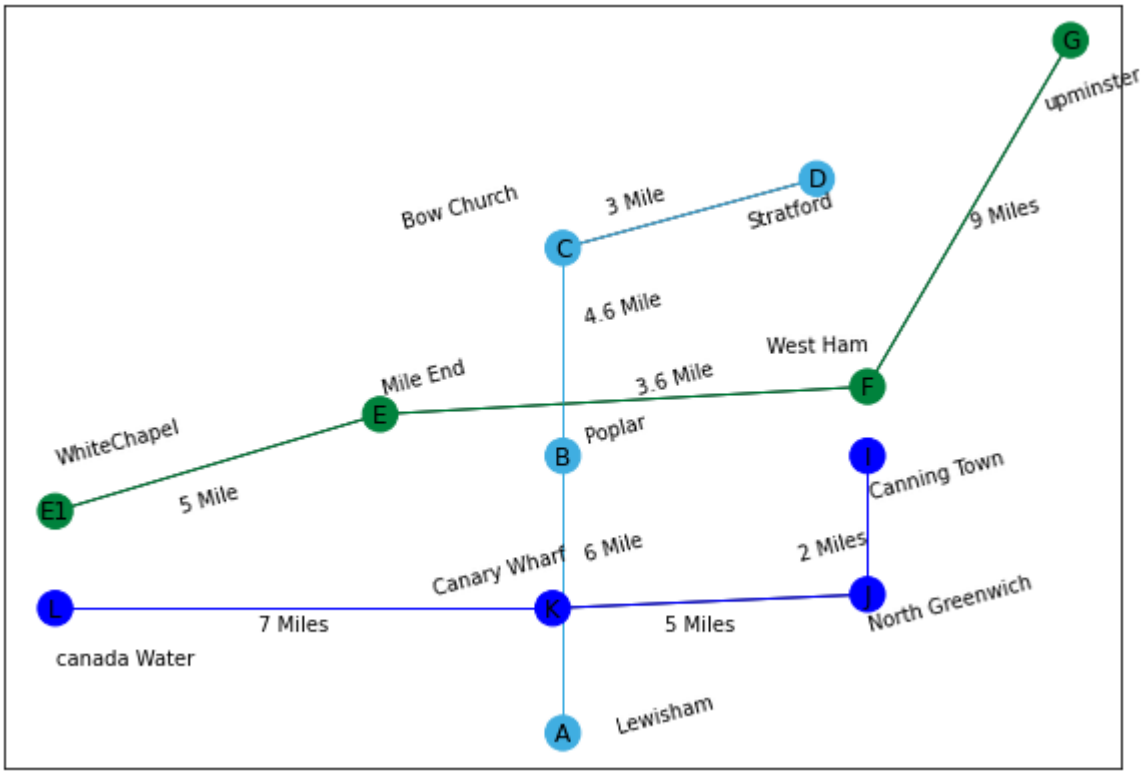
```
M: [[5.]
 [7.]
 [4.]]
```

week 4

In [350…

```
# Create a graph object

MyGraph = nx.Graph()

# Add nodes

# Blue Nodes
MyGraph.add_node('A', npos=(50, 10), ccn='#40AEE1')
MyGraph.add_node('B', npos=(50, 50), ccn='#40AEE1')
MyGraph.add_node('C', npos=(50, 80), ccn='#40AEE1')
MyGraph.add_node('D', npos=(75, 90), ccn='#40AEE1')

# Green Nodes
MyGraph.add_node('E1', npos=(0, 42), ccn='#00823C')
MyGraph.add_node('E', npos=(32, 56), ccn='#00823C')
MyGraph.add_node('F', npos=(80, 60), ccn='#00823C')
MyGraph.add_node('G', npos=(100, 110), ccn='#00823C')

# Purple Nodes
MyGraph.add_node('I', npos=(80, 50), ccn='#0000FF')
MyGraph.add_node('J', npos=(80, 30), ccn='#0000FF')
MyGraph.add_node('K', npos=(49, 28), ccn='#0000FF')
MyGraph.add_node('L', npos=(0, 28), ccn='#0000FF')

# Connect nodes
# Blue Line
MyGraph.add_edge('A', 'B', cce='#40AEE1')
MyGraph.add_edge('B', 'C', cce='#40AEE1')
MyGraph.add_edge('C', 'D', cce='#40AEE1')

# Green Line
MyGraph.add_edge('E1', 'E', cce='#00823C')
MyGraph.add_edge('E', 'F', cce='#00823C')
MyGraph.add_edge('F', 'G', cce='#00823C')

# Purple Line
MyGraph.add_edge('I', 'J', cce='#0000FF')
MyGraph.add_edge('J', 'K', cce='#0000FF')
MyGraph.add_edge('K', 'L', cce='#0000FF')
```

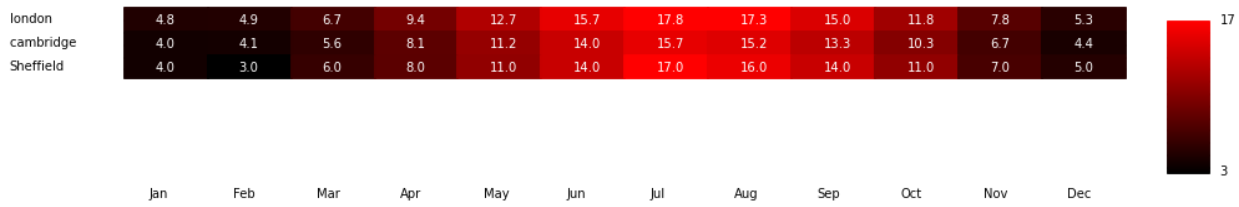
```
# Extract attributes from the graph to dictionaries
pos = nx.get_node_attributes(MyGraph, 'npos')
nodecolour = nx.get_node_attributes(MyGraph, 'ccn')
edgecolour = nx.get_edge_attributes(MyGraph, 'cce')

# Place the dictionary values in lists
NodeList = list(nodecolour.values())
EdgeList = list(edgecolour.values())

# Set the size of the figure
plt.figure(figsize=(10, 7))

# Display the names of the stations
plt.text(55, 10, s='Lewisham', rotation=15)
plt.text(52, 35, s='6 Mile', rotation=15)
plt.text(52, 52, s='Poplar', rotation=15)
plt.text(52, 69, s='4.6 Mile', rotation=15)
plt.text(34, 83, s='Bow Church', rotation=15)
plt.text(54, 85, s='3 Mile', rotation=15)
plt.text(68, 83, s='Stratford', rotation=15)

plt.text(0, 49, s='WhiteChapel', rotation=15)
plt.text(12, 42, s='5 Mile', rotation=15)
plt.text(32, 59, s='Mile End', rotation=15)
plt.text(57, 59, s='3.6 Mile', rotation=15)
plt.text(70, 65, s='West Ham', rotation=1)
plt.text(90, 83, s='9 Miles', rotation=15)
plt.text(97, 100, s='upminster', rotation=18)

plt.text(80, 44, s='Canning Town', rotation=15)
plt.text(73, 35, s='2 Miles', rotation=15)
plt.text(80, 25, s='North Greenwich', rotation=15)
plt.text(60, 25, s='5 Miles', rotation=0)
plt.text(37, 30, s='Canary Wharf', rotation=15)
plt.text(20, 25, s='7 Miles', rotation=0)
plt.text(0, 20, s='canada Water', rotation=0)

# Draw the nodes and the edges
nx.draw_networkx(MyGraph, pos, node_color=NodeList)
nx.draw_networkx_edges(MyGraph, pos, edge_color=EdgeList)

# Visualise the graph
plt.show()
```



In [357…

```
## part 2

# This subroutine encapsulates the 'plot' method, as the most suitable for raster rende
def DrawBox(x, y, size, r, g, b):
    if r < 0:
        r = int(0)
    if g < 0:
        g = int(0)
    if b < 0:
        b = int(0)
    if r > 255:
        r = int(255)
    if g > 255:
        g = int(255)
    if b > 255:
        b = int(255)
    for i in range(0, int(size)):
        plt.plot([x, x + size], [y + i, y + i], '#{:02x}{:02x}{:02x}'.format(r, g, b))

df = pd.read_csv(r'HeatMap.csv')
# Set the plot
plt.figure(figsize=(18, 3.5))
plt.axis([0, 600, 0, 400])
plt.xticks([])
plt.yticks([])
plt.axis('off')

Min = int(min(df.min(numeric_only=True)))
Max = int(max(df.max(numeric_only=True)))

BoxSize = int(40)
OffsetX = int(15)
OffsetY = int(12)
```

```
# Generate the heat map
for i in range(0, df.shape[0]):
    for j in range(1, df.shape[1]):
        ColourCode = int(((df.values[i, j]-Min)/(Max-Min))*255)
        DrawBox(20+BoxSize*j, 300-BoxSize*i, BoxSize, ColourCode, 0, 0)
        plt.text(OffsetX+20+BoxSize*j, OffsetY+300-BoxSize*i, str(df.values[i, j]), col

for i in range(0, 256):
    plt.plot([560, 580], [i + 60, i + 60], '#{:02x}{:02x}{:02x}'.format(int(i), 0, 0))

plt.text(585, 58, Min)
plt.text(585, 312, Max)

plt.text(72, 20, 'Jan')
plt.text(112, 20, 'Feb')
plt.text(152, 20, 'Mar')
plt.text(192, 20, 'Apr')
plt.text(232, 20, 'May')
plt.text(272, 20, 'Jun')
plt.text(312, 20, 'Jul')
plt.text(352, 20, 'Aug')
plt.text(392, 20, 'Sep')
plt.text(432, 20, 'Oct')
plt.text(472, 20, 'Nov')
plt.text(512, 20, 'Dec')

plt.text(5, 315, str(df.values[0, 0]))
plt.text(5, 275, str(df.values[1, 0]))
plt.text(5, 235, str(df.values[2, 0]))


plt.show()
```

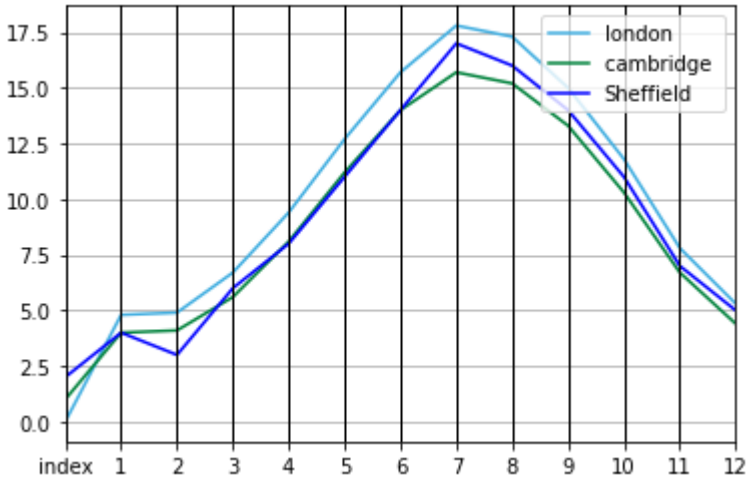


In [363…

```
## Part 3
print(df.to_string())
pd.plotting.parallel_coordinates( df.reset_index(), 'City', color=('#40AEE1', '#00823C'
```

```
        City    1    2    3    4     5     6     7     8     9    10   11   12
0     london  4.8  4.9  6.7  9.4  12.7  15.7  17.8  17.3  15.0  11.8  7.8  5.3
1  cambridge  4.0  4.1  5.6  8.1  11.2  14.0  15.7  15.2  13.3  10.3  6.7  4.4
2  Sheffield  4.0  3.0  6.0  8.0  11.0  14.0  17.0  16.0  14.0  11.0  7.0  5.0
```

Out[363… <AxesSubplot:>


17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
index 1 2 3 4 5 6 7 8 9 10 11 12
london
cambridge
Sheffield

In [ ]: