Assignment - 1

---

1. Import Necessary Libraires

In [1]:
```
import pandas as pd
import numpy as np
import statistics
import seaborn as sns
from scipy import stats
from matplotlib import pyplot as plt

import warnings
warnings.filterwarnings('ignore')
```

Q 7) Calculate Mean, Median, Mode, Variance, Standard Deviation, Range & comment about the values / draw inferences, for the given dataset

In [2]:
```
q7 = pd.read_csv('Q7.csv')
q7
```

Out[2]:

	Unnamed: 0	Points	Score	Weigh
0	Mazda RX4	3.90	2.620	16.46
1	Mazda RX4 Wag	3.90	2.875	17.02
2	Datsun 710	3.85	2.320	18.61
3	Hornet 4 Drive	3.08	3.215	19.44
4	Hornet Sportabout	3.15	3.440	17.02
5	Valiant	2.76	3.460	20.22
6	Duster 360	3.21	3.570	15.84
7	Merc 240D	3.69	3.190	20.00
8	Merc 230	3.92	3.150	22.90
9	Merc 280	3.92	3.440	18.30
10	Merc 280C	3.92	3.440	18.90
11	Merc 450SE	3.07	4.070	17.40
12	Merc 450SL	3.07	3.730	17.60
13	Merc 450SLC	3.07	3.780	18.00
14	Cadillac Fleetwood	2.93	5.250	17.98
15	Lincoln Continental	3.00	5.424	17.82
16	Chrysler Imperial	3.23	5.345	17.42
17	Fiat 128	4.08	2.200	19.47
18	Honda Civic	4.93	1.615	18.52
19	Toyota Corolla	4.22	1.835	19.90
20	Toyota Corona	3.70	2.465	20.01
21	Dodge Challenger	2.76	3.520	16.87
22	AMC Javelin	3.15	3.435	17.30
23	Camaro Z28	3.73	3.840	15.41
24	Pontiac Firebird	3.08	3.845	17.05
25	Fiat X1-9	4.08	1.935	18.90
26	Porsche 914-2	4.43	2.140	16.70
27	Lotus Europa	3.77	1.513	16.90
28	Ford Pantera L	4.22	3.170	14.50
29	Ferrari Dino	3.62	2.770	15.50
30	Maserati Bora	3.54	3.570	14.60
31	Volvo 142E	4.11	2.780	18.60

Calculate Mean, Median, Mode, Variance, Standard Deviation, Range for 'Points'

In [3]:
```
print('Mean of Points               :',round(np.mean(q7['Points']),4))
print('Median of Points             :',round(np.median(q7['Points']),4))
print('Mode of Points               :',statistics.mode(q7['Points']))
print('Multimode of Points          :',statistics.multimode(q7['Points']))
print('Variance of Points           :',round(np.var(q7['Points']),4))
print('Standard Deviation of Points :',round(np.std(q7['Points']),4))
print('Range of Points              :', np.min(q7['Points']) ,'to', np.max(q7['Points']))
```

```
Mean of Points               : 3.5966
Median of Points             : 3.695
Mode of Points               : 3.92
Multimode of Points          : [3.92, 3.07]
Variance of Points           : 0.2769
Standard Deviation of Points : 0.5263
Range of Points              : 2.76 to 4.93
```

Calculate Mean, Median, Mode, Variance, Standard Deviation, Range for 'Score'

In [4]:
```
print('Mean of Score               :',round(np.mean(q7['Score']),4))
print('Median of Score             :',round(np.median(q7['Score']),4))
print('Mode of Score               :',statistics.mode(q7['Score']))
print('Variance of Score           :',round(np.var(q7['Score']),4))
print('Standard Deviation of Score :',round(np.std(q7['Score']),4))
print('Range of Score              :', np.min(q7['Score']) ,'to', np.max(q7['Score']))
```

```
Mean of Score               : 3.2172
Median of Score             : 3.325
Mode of Score               : 3.44
Variance of Score           : 0.9275
Standard Deviation of Score : 0.963
Range of Score              : 1.513 to 5.424
```

Calculate Mean, Median, Mode, Variance, Standard Deviation, Range for 'Weigh'

In [5]:
```
print('Mean of Weigh               :',round(np.mean(q7['Weigh']),4))
print('Median of Weigh             :',round(np.median(q7['Weigh']),4))
print('Mode of Weigh               :',statistics.mode(q7['Weigh']))
print('Multimode of Weigh          :',statistics.multimode(q7['Weigh']))
print('Variance of Weigh           :',round(np.var(q7['Weigh']),4))
print('Standard Deviation of Weigh :',round(np.std(q7['Weigh']),4))
print('Range of Weigh              :', np.min(q7['Weigh']) ,'to', np.max(q7['Weigh']))
```

```
Mean of Weigh               : 17.8488
Median of Weigh             : 17.71
Mode of Weigh               : 17.02
Multimode of Weigh          : [17.02, 18.9]
Variance of Weigh           : 3.0934
Standard Deviation of Weigh : 1.7588
Range of Weigh              : 14.5 to 22.9
```

Q9) Calculate Skewness, Kurtosis & draw inferences on the following data Cars speed and distance

In [6]:
```
q9_a = pd.read_csv('Q9_a.csv')
q9_a
```

Out[6]:

	Index	speed	dist
0	1	4	2
1	2	4	10
2	3	7	4
3	4	7	22
4	5	8	16
5	6	9	10
6	7	10	18
7	8	10	26
8	9	10	34
9	10	11	17
10	11	11	28
11	12	12	14
12	13	12	20
13	14	12	24
14	15	12	28
15	16	13	26
16	17	13	34
17	18	13	34
18	19	13	46
19	20	14	26
20	21	14	36
21	22	14	60
22	23	14	80
23	24	15	20
24	25	15	26
25	26	15	54
26	27	16	32
27	28	16	40
28	29	17	32
29	30	17	40
30	31	17	50
31	32	18	42
32	33	18	56
33	34	18	76
34	35	18	84
35	36	19	36
36	37	19	46
37	38	19	68
38	39	20	32
39	40	20	48
40	41	20	52
41	42	20	56
42	43	20	64
43	44	22	66
44	45	23	54
45	46	24	70
46	47	24	92
47	48	24	93
48	49	24	120
49	50	25	85

Calculate Skewness & Kurtosis for speed

In [7]:
```
print('Skewness of speed :',round(q9_a['speed'].skew(),4))
print('Kurtosis of speed :',round(q9_a['speed'].kurt(),4))
```

```
Skewness of speed : -0.1175
Kurtosis of speed : -0.509
```

Inference:The distribution of speed is slightly negatively skewed and playkurtic.

Calculate Skewness & Kurtosis for distance

In [8]:
```
print('Skewness of distance :',round(q9_a['dist'].skew(),4))
print('Kurtosis of distance :',round(q9_a['dist'].kurt(),4))
```

```
Skewness of distance : 0.8069
Kurtosis of distance : 0.4051
```

Inference: The distribution of speed is slightly positively skewed and playkurtic.

In [9]:
```
q9_b = pd.read_csv('Q9_b.csv')
q9_b
```

Out[9]:

	Unnamed: 0	SP	WT
0	1	104.185353	28.762059
1	2	105.461264	30.466833
2	3	105.461264	30.193597
3	4	113.461264	30.632114
4	5	104.461264	29.889149
...	...	...	...
76	77	169.598513	16.132947
77	78	150.576579	37.923113
78	79	151.598513	15.769625
79	80	167.944460	39.423099
80	81	139.840817	34.948615

81 rows × 3 columns

Calculate Skewness & Kurtosis for SP

In [10]:
```
print('Skewness of SP :',round(q9_b['SP'].skew(),4))
print('Kurtosis of SP :',round(q9_b['SP'].kurt(),4))
```

```
Skewness of SP : 1.6115
Kurtosis of SP : 2.9773
```

Inference: The distribution of speed is positively skewed and playkurtic.

Calculate Skewness & Kurtosis for WT

In [11]:
```
print('Skewness of WT :',round(q9_b['WT'].skew(),4))
print('Kurtosis of WT :',round(q9_b['WT'].kurt(),4))
```

```
Skewness of WT : -0.6148
Kurtosis of WT : 0.9503
```

Inference: The distribution of speed is slightly negatively skewed and playkurtic.

--

Q.11 Suppose we want to estimate the average weight of an adult male in Mexico. We draw a random sample of 2,000 men from a population of 3,000,000 men and weigh them. We find that the average person in our sample weighs 200 pounds, and the standard deviation of the sample is 30 pounds. Calculate 94%, 98%, 96% confidence interval?

In [12]:
```
ci_94 = stats.norm.interval(alpha=0.94, loc=200, scale=(30/pow(2000,0.5)))
print('The average weight of an adult in Mexico with 94% Confidence Interval is: ',np.round(ci_94,4))
```

```
The average weight of an adult in Mexico with 94% Confidence Interval is:  [198.7383 201.2617]
```

In [13]:
```
ci_98 = stats.norm.interval(alpha=0.98, loc=200, scale=(30/pow(2000,0.5)))
print('The average weight of an adult in Mexico with 98% Confidence Interval is: ',np.round(ci_98,4))
```

```
The average weight of an adult in Mexico with 98% Confidence Interval is:  [198.4394 201.5606]
```

In [14]:
```
ci_96 = stats.norm.interval(alpha=0.96, loc=200, scale=(30/pow(2000,0.5)))
print('The average weight of an adult in Mexico with 96% Confidence Interval is: ',np.round(ci_96,4))
```

```
The average weight of an adult in Mexico with 96% Confidence Interval is:  [198.6223 201.3777]
```

Cross checking the answers of Q.12 with python code

In [15]:
```
book_2 = pd.read_excel('Book2.xlsx')
book_2
```

Out[15]:

	values
0	34
1	36
2	36
3	38
4	38
5	39
6	39
7	40
8	40
9	41
10	41
11	41
12	41
13	42
14	42
15	45
16	49
17	56

In [16]:
```
np.mean(book_2)
```

Out[16]:
```
values    41.0
dtype: float64
```

In [17]:
```
np.median(book_2)
```

Out[17]:
```
40.5
```

In [18]:
```
statistics.mode(book_2['values'])
```

Out[18]:
```
41
```

In [19]:
```
np.var(book_2)
```

Out[19]:
```
values    24.111111
dtype: float64
```

In [20]:
```
np.std(book_2)
```

Out[20]:
```
values    4.910307
dtype: float64
```

Q 20) Calculate probability from the given dataset for the below cases

```
a.  P(MPG>38)
b.  P(MPG<40)
c.  P(20<MPG<50)
```

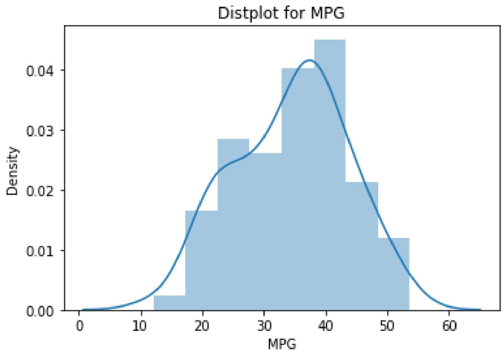
In [21]:
```
cars_data = pd.read_csv('Cars.csv')
cars_data
```

Out[21]:

	HP	MPG	VOL	SP	WT
0	49	53.700681	89	104.185353	28.762059
1	55	50.013401	92	105.461264	30.466833
2	55	50.013401	92	105.461264	30.193597
3	70	45.696322	92	113.461264	30.632114
4	53	50.504232	92	104.461264	29.889149
...	...	...	...	...	...
76	322	36.900000	50	169.598513	16.132947
77	238	19.197888	115	150.576579	37.923113
78	263	34.000000	50	151.598513	15.769625
79	295	19.833733	119	167.944460	39.423099
80	236	12.101263	107	139.840817	34.948615

81 rows × 5 columns

In [56]:
```
sns.distplot(a=cars_data['MPG'])
plt.title('Distplot for MPG')
plt.show()
```



a. P (MPG>38) - Area to the right

In [59]:
```
1-stats.norm.cdf(38,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std())
```

Out[59]: 0.3475939251582705

In [62]:
```
print('Probability of MPG > 38 is', round(1-stats.norm.cdf(38,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std()),4))
```

Probability of MPG > 38 is 0.3476

b. P (MPG<40) - Area to the left

In [60]:
```
stats.norm.cdf(40,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std())
```

Out[60]: 0.7293498762151616

In [61]:
```
print('Probability of MPG < 40 is', round(stats.norm.cdf(40,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std()),4))
```

Probability of MPG < 40 is 0.7293

c. P (20<MPG<50) - P(MPG<50)-P(MPG<20)

In [64]:
```
stats.norm.cdf(50,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std())
-stats.norm.cdf(20,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std())
```

Out[64]: -0.05712377632115936

In [65]:
```
print('Probability of 20 > MPG > 50 is', round(stats.norm.cdf(50,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std())
-stats.norm.cdf(20,loc=cars_data['MPG'].mean(),scale=cars_data['MPG'].std()),4))
```

Probability of 20 > MPG > 50 is 0.8989

--

Q 21) Check whether the data follows normal distribution

```
a.  Check whether the MPG of Cars follows Normal Distribution [Dataset: Cars.csv]
b.  Check Whether the Adipose Tissue (AT) and Waist Circumference (Waist) from wc-at
    data set follows Normal Distribution [Dataset: wc-at.csv]
```

a. Check whether the MPG of Cars follows Normal Distribution

In [30]:
```
np.mean(cars_data['MPG'])
```

Out[30]: 34.422075728024666

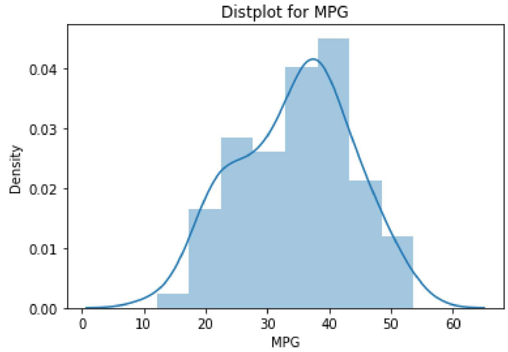
In [31]:
```
np.median(cars_data['MPG'])
```

Out[31]: 35.15272697

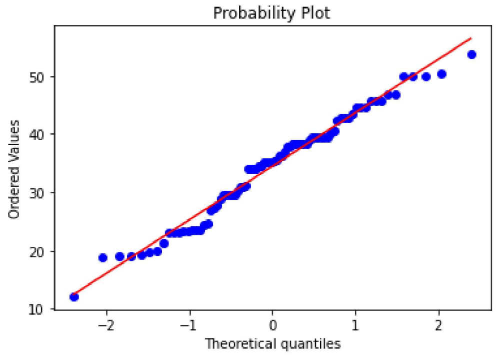
In [32]:
```
statistics.mode(cars_data['MPG'])
```

Out[32]: 29.62993595

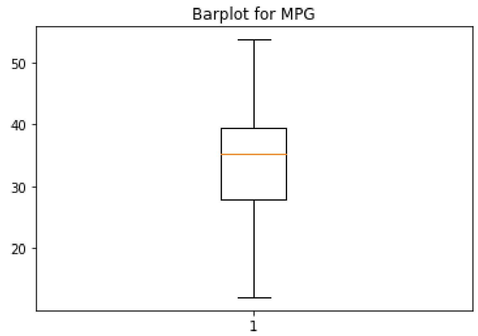
In [33]:
```
sns.distplot(a=cars_data['MPG'])
plt.title('Distplot for MPG')
plt.show()
```



In [34]:
```
stats.probplot(x=cars_data['MPG'], dist='norm', plot=plt)
plt.show()
```



In [35]:
```
plt.boxplot(x='MPG', data=cars_data)
plt.title('Barplot for MPG')
plt.show()
```



In [36]:
```
round(cars_data['MPG'].skew(),4)
```

Out[36]: -0.1779

In [37]:
```
round(cars_data['MPG'].kurtosis(),4)
```
Out[37]: -0.6117

b. Check Whether the Adipose Tissue (AT) and Waist Circumference (Waist) from wc-at data set follows Normal Distribution

In [38]:
```
wc_at_data = pd.read_csv('wc-at.csv')
wc_at_data
```
Out[38]:

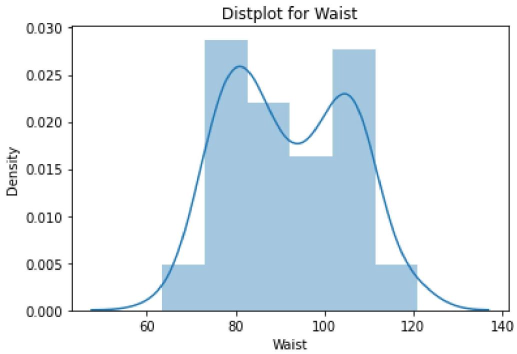
	Waist	AT
0	74.75	25.72
1	72.60	25.89
2	81.80	42.60
3	83.95	42.80
4	74.65	29.84
...	...	...
104	100.10	124.00
105	93.30	62.20
106	101.80	133.00
107	107.90	208.00
108	108.50	208.00

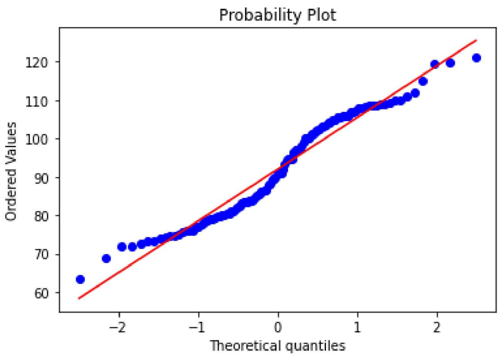
109 rows × 2 columns

In [39]:
```
np.mean(wc_at_data['Waist'])
```
Out[39]: 91.90183486238533

In [40]:
```
np.median(wc_at_data['Waist'])
```
Out[40]: 90.8

In [41]:
```
statistics.mode(wc_at_data['Waist'])
```
Out[41]: 94.5

In [42]:
```
sns.distplot(a=wc_at_data['Waist'])
plt.title('Distplot for Waist')
plt.show()
```


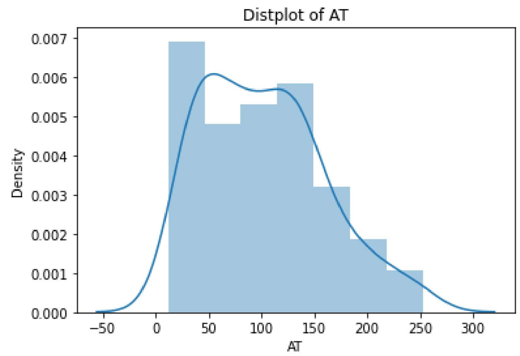
In [43]:
```
stats.probplot(x=wc_at_data['Waist'], dist='norm', plot=plt)
plt.show()
```


In [44]:
```
np.mean(wc_at_data['AT'])
```
Out[44]: 101.89403669724771

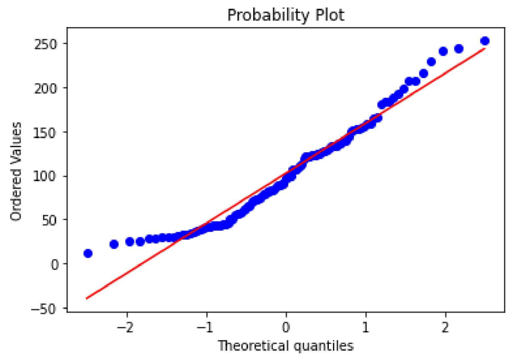
In [45]:
```
np.median(wc_at_data['AT'])
```
Out[45]: 96.54

In [46]:
```
statistics.mode(wc_at_data['AT'])
```
Out[46]: 121.0

In [47]:
```
sns.distplot(wc_at_data['AT'])
plt.title('Distplot of AT')
plt.show()
```



In [48]:
```
stats.probplot(x=wc_at_data['AT'], dist='norm', plot=plt)
plt.show()
```



Q 22) Calculate the Z scores of 90% confidence interval,94% confidence interval, 60% confidence interval

Z scores by PPF

In [49]:
```
z_90 = stats.norm.ppf(0.95) #z score for 90
print('Z score of 90% confidence interval is:', round(z_90,4))
```
```
Z score of 90% confidence interval is: 1.6449
```

In [70]:
```
z_94 = stats.norm.ppf(0.97) #z score for 94
print('Z score of 94% confidence interval is:', round(z_94,4))
```
```
Z score of 94% confidence interval is: 1.8808
```

In [69]:
```
z_60 = stats.norm.ppf(0.80) #z score for 60
print('Z score of 60% confidence interval is:', round(z_60,4))
```
```
Z score of 60% confidence interval is: 0.8416
```

Z scores by Interval

In [66]:
```
stats.norm.interval(0.90,loc=0,scale=1) #z score for 90
```
Out[66]: (-1.6448536269514729, 1.6448536269514722)

In [67]:
```
stats.norm.interval(0.94,loc=0,scale=1) #z score for 94
```
Out[67]:
```
(-1.8807936081512509, 1.8807936081512509)
```

In [68]:
```
stats.norm.interval(0.6,loc=0,scale=1) #z score for 60
```
Out[68]:
```
(-0.8416212335729142, 0.8416212335729143)
```

Q 23) Calculate the t scores of 95% confidence interval, 96% confidence interval, 99% confidence interval for sample size of 25

t scores by PPF

In [52]:
```
t_95 = stats.t.ppf(q=0.975, df=24)
print('t score of 95% confidence interval is:',round(t_95,4))
```
```
t score of 95% confidence interval is: 2.0639
```

In [53]:
```
t_96 = stats.t.ppf(q=0.98, df=24)
print('t score of 96% confidence interval is:',round(t_96,4))
```
```
t score of 96% confidence interval is: 2.1715
```

In [54]:
```
t_99 = stats.t.ppf(q=0.995, df=24)
print('t score of 99% confidence interval is:',round(t_99,4))
```
```
t score of 99% confidence interval is: 2.7969
```

t scores by Interval

In [71]:
```
# t scores of 95% CI
stats.t.interval(0.95,25,loc=0,scale=1)
```
Out[71]:
```
(-2.059538552753294, 2.059538552753294)
```

In [72]:
```
# t scores of 96% CI
stats.t.interval(0.96,25,loc=0,scale=1)
```
Out[72]:
```
(-2.1665866344527562, 2.1665866344527562)
```

In [73]:
```
# t scores of 99% CI
stats.t.interval(0.99,25,loc=0,scale=1)
```
Out[73]:
```
(-2.787435813675851, 2.787435813675851)
```