Q1) Identify the Data type for the Following:

Activity	Data Type
Number of beatings from Wife	Discrete (Countable datatype)
Results of rolling a dice	Discrete (Countable datatype)
Weight of a person	Continuous data type
Weight of Gold	Continuous data type
Distance between two places	Continuous data type
Length of a leaf	Continuous data type
Dog's weight	Continuous data type
Blue Color	Classification (Nominal)data type
Number of kids	Countable data type
Number of tickets in Indian railways	Discrete (Countable data type)
Number of times married	Discrete (Countable data type)
Gender (Male or Female)	Classification (Nominal) data type

Q2) Identify the Data types, which were among the following

Nominal, Ordinal, Interval, Ratio.

Data	Data Type
Gender	Nominal
High School Class Ranking	Ordinal
Celsius Temperature	Interval
Weight	Ratio
Hair Color	Nominal
Socioeconomic Status	Ordinal
Fahrenheit Temperature	Interval
Height	Ratio
Type of living accommodation	Nominal
Level of Agreement	Ordinal
IQ(Intelligence Scale)	Ratio
Sales Figures	Ratio
Blood Group	Nominal
Time Of Day	Nominal
Time on a Clock with Hands	Ordinal
Number of Children	Ratio
Religious Preference	Nominal

Barometer Pressure	Interval
SAT Scores	Interval
Years of Education	Ordinal

Q3) Three Coins are tossed, find the probability that two heads and one tail are obtained?

Total number of possible outcomes when tossing three coins :As each coin has two possible outcomes (head/tail) since there are three coins the total outcomes are 2^3 or 2*2*2=8 outcomes which comes as follows

HHH, HHT, HTH, THH, TTT, TTH, THT, HTT→n(s)=8

n(a)=event occurring two heads and one tail as there is no specific condition mentioned we have three outcomes in our favour (THH, HTH, HHT) =3

Total probability=n(a)/n(s)=3/8.

Q4) Two Dice are rolled, find the probability that sum is

a) Equal to 1
b) Less than or equal to 4
c) Sum is divisible by 2 and 3

When two dice are rolled ,as each die has 6 different outcomes (1,2,3,4,5,6) the total number of outcomes are 6*6=36 possibilities. The possibilities are
(1,1),(1,2),(1,3),(1,4),(1,5),(1,6)
(2,1),(2,2),(2,3),(2,4),(2,5),(2,6)
(3,1),(3,2),(3,3),(3,4),(3,5),(3,6)
(4,1),(4,2),(4,3),(4,4),(4,5),(4,6)
(5,1),(5,2),(5,3),(5,4),(5,5),(5,6)
(6,1),(6,2),(6,3),(6,4),(6,5),(6,6)
a)As there is no possibility of occurring the sum is equal to 1 when throwing two dice ,the probability is considered as 0.It is possible as the probability always lies between **0 <=P(A)<=1.**

b)sum is less than or equal to 4
total no of possibilities occurring in our favour
n(a)=6{(1,1),(1,2),(1,3),(2,1),(2,2),(3,1)}
total number of possibilities n(s) =36
total probability=n(a)/n(s)=6/36=1/6.

c)sum is divisible by 2 and 3
so here as the sum is said to be divisible by 2 &3 it is also divisible by 6 this states divisibility rule of 6 so now we have to check the ordered pairs sum which is divisible by 6 so the possibilities of outcome which are in favour are n(a)=6{(1,5),(2,4),(3,3),(4,2),(5,1),(6,6)}
total possibilities n(s)=36
required probability=n(a)/n(s)=6/36=1/6.

Q5) A bag contains 2 red, 3 green and 2 blue balls. Two balls are drawn at random. What is the probability that none of the balls drawn is blue?

Total number of balls=>2+3+2=7

So we have to select the two balls such that none of the balls are blue in color so apart the blue color balls we have red and green balls numbered as 2&3 respectively in total 5 balls so we can select two non blue balls from above 5 balls which is the favourable condition for the above problem

Total probability==$^5C_2$ $/^7{}_{\text{C2= [5!/(5-2)}}$!*2!]/[7!/(7-2)!*2!]= [5*4/2*1]/[7*6/2*1]= 10/21=0.476

Q6) Calculate the Expected number of candies for a randomly selected child

Below are the probabilities of count of candies for children (ignoring the nature of the child-Generalized view)

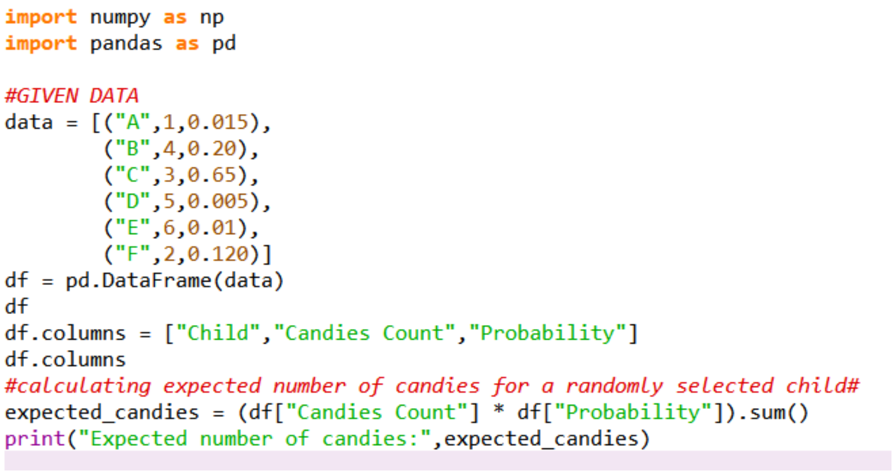
CHILD	Candies count	Probability
A	1	0.015
B	4	0.20
C	3	0.65
D	5	0.005
E	6	0.01
F	2	0.120

Child A – probability of having 1 candy = 0.015.

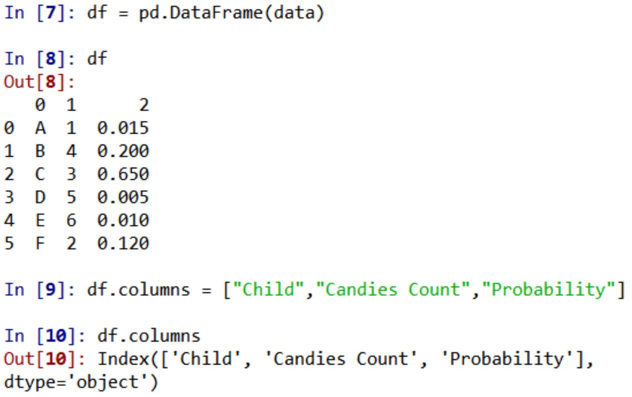
Child B – probability of having 4 candies = 0.20

Expected Value=∑(Value× Probability)

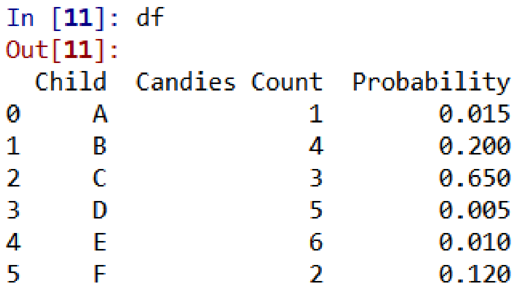
where value here is Candies count and Probability is equal to probability of having number of candies by each individual .this can be achieved by pandas. Firstly create a numpy data by importing numpy library and change numpy data into pandas and assign the column names to each column and apply the formula over the pandas data.Here is the code for above question



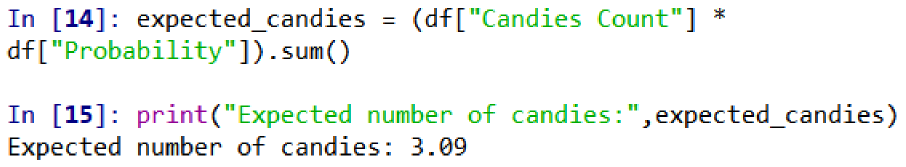
```
import numpy as np
import pandas as pd

#GIVEN DATA
data = [("A",1,0.015),
        ("B",4,0.20),
        ("C",3,0.65),
        ("D",5,0.005),
        ("E",6,0.01),
        ("F",2,0.120)]
df = pd.DataFrame(data)
df
df.columns = ["Child","Candies Count","Probability"]
df.columns
#calculating expected number of candies for a randomly selected child#
expected_candies = (df["Candies Count"] * df["Probability"]).sum()
print("Expected number of candies:",expected_candies)
```



```
In [7]: df = pd.DataFrame(data)

In [8]: df
Out[8]:
   0  1      2
0  A  1  0.015
1  B  4  0.200
2  C  3  0.650
3  D  5  0.005
4  E  6  0.010
5  F  2  0.120

In [9]: df.columns = ["Child","Candies Count","Probability"]

In [10]: df.columns
Out[10]: Index(['Child', 'Candies Count', 'Probability'],
dtype='object')
```



```
In [11]: df
Out[11]:
  Child  Candies Count  Probability
0     A              1        0.015
1     B              4        0.200
2     C              3        0.650
3     D              5        0.005
4     E              6        0.010
5     F              2        0.120
```



```
In [14]: expected_candies = (df["Candies Count"] *
df["Probability"]).sum()

In [15]: print("Expected number of candies:",expected_candies)
Expected number of candies: 3.09
```

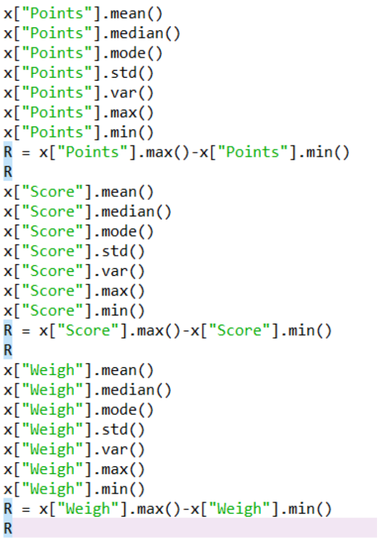
Q7) Calculate Mean, Median, Mode, Variance, Standard Deviation, Range & comment about the values / draw inferences, for the given dataset

- For Points,Score,Weigh>
  Find Mean, Median, Mode, Variance, Standard Deviation, and Range and also Comment about the values/ Draw some inferences.

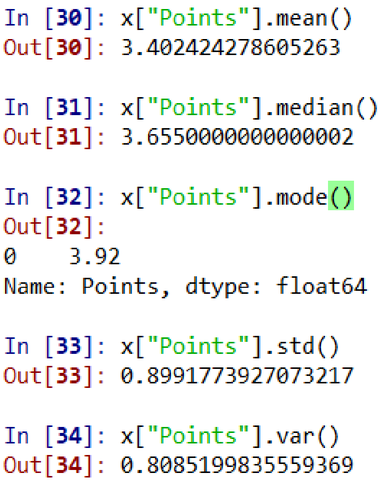
Use Q7.csv file

Name	Points	Score	Weigh
Mazda RX4	3.9	2.62	16.46
Mazda RX4 Wag	3.9	2.875	17.02
Datsun 710	3.85	2.32	18.61
Hornet 4 Drive	3.08	3.215	19.44
Hornet Sportabout	3.15	3.44	17.02
Valiant	2.76	3.46	20.22
Duster 360	3.21	3.57	15.84
Merc 240D	3.69	3.19	20
Merc 230	3.92	3.15	22.9
Merc 280	3.92	3.44	18.3
Merc 280C	3.92	3.44	18.9
Merc 450SE	3.07	4.07	17.4
Merc 450SL	3.07	3.73	17.6
Merc 450SLC	3.07	3.78	18
Cadillac Fleetwood	2.93	5.25	17.98
Lincoln Continental	3	5.424	17.82
Chrysler Imperial	3.23	5.345	17.42
Fiat 128	4.08	2.2	19.47
Honda Civic	4.93	1.615	18.52
Toyota Corolla	4.22	1.835	19.9
Toyota Corona	3.7	2.465	20.01

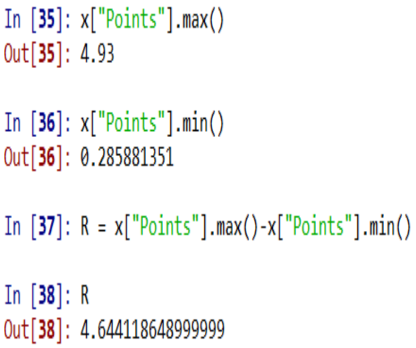
Dodge Challenger	2.76	3.52	16.87
AMC Javelin	3.15	3.435	17.3
Camaro Z28	3.73	3.84	15.41
Pontiac Firebird	3.08	3.845	17.05
Fiat X1-9	4.08	1.935	18.9
Porsche 914-2	4.43	2.14	16.7
Lotus Europa	3.77	1.513	16.9
Ford Pantera L	4.22	3.17	14.5
Ferrari Dino	3.62	2.77	15.5
Maserati Bora	3.54	3.57	14.6
Volvo 142E	4.11	2.78	18.6
Mean	3.596563	3.21725	17.84875
Median	3.695	3.325	17.71
Mode	3.92	3.44	17.02
Standard Deviation	0.534679	0.978457	1.786943
Variance	0.285881	0.957379	3.193166
Range	2.17	3.911	8.4



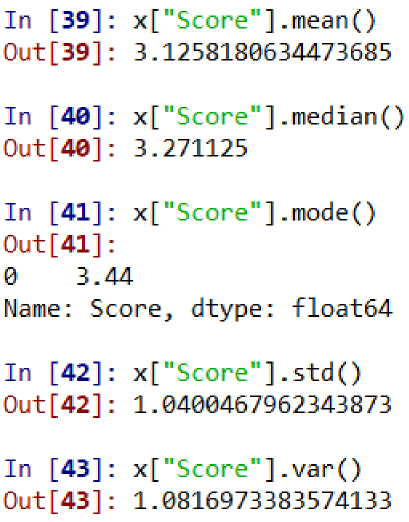
```
x["Points"].mean()
x["Points"].median()
x["Points"].mode()
x["Points"].std()
x["Points"].var()
x["Points"].max()
x["Points"].min()
R = x["Points"].max()-x["Points"].min()
R
x["Score"].mean()
x["Score"].median()
x["Score"].mode()
x["Score"].std()
x["Score"].var()
x["Score"].max()
x["Score"].min()
R = x["Score"].max()-x["Score"].min()
R
x["Weigh"].mean()
x["Weigh"].median()
x["Weigh"].mode()
x["Weigh"].std()
x["Weigh"].var()
x["Weigh"].max()
x["Weigh"].min()
R = x["Weigh"].max()-x["Weigh"].min()
R
```



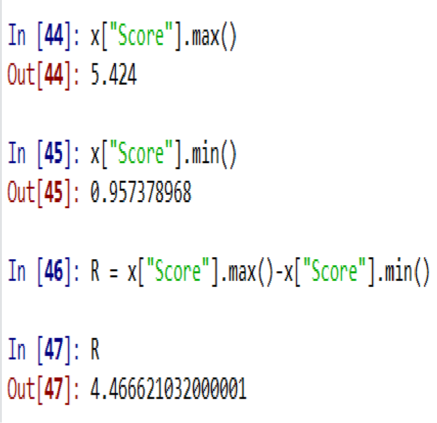
```
In [30]: x["Points"].mean()
Out[30]: 3.402424278605263

In [31]: x["Points"].median()
Out[31]: 3.6550000000000002

In [32]: x["Points"].mode()
Out[32]:
0    3.92
Name: Points, dtype: float64

In [33]: x["Points"].std()
Out[33]: 0.8991773927073217

In [34]: x["Points"].var()
Out[34]: 0.8085199835559369
```



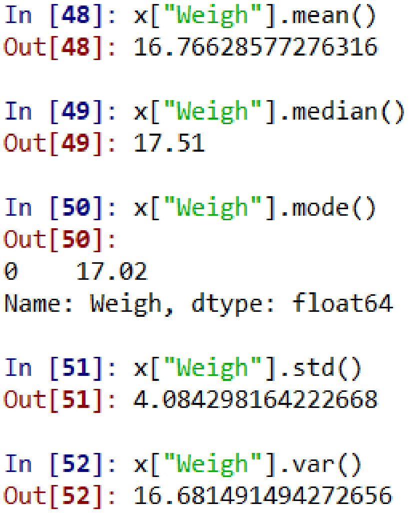
```
In [35]: x["Points"].max()
Out[35]: 4.93

In [36]: x["Points"].min()
Out[36]: 0.285881351

In [37]: R = x["Points"].max()-x["Points"].min()

In [38]: R
Out[38]: 4.644118648999999
```



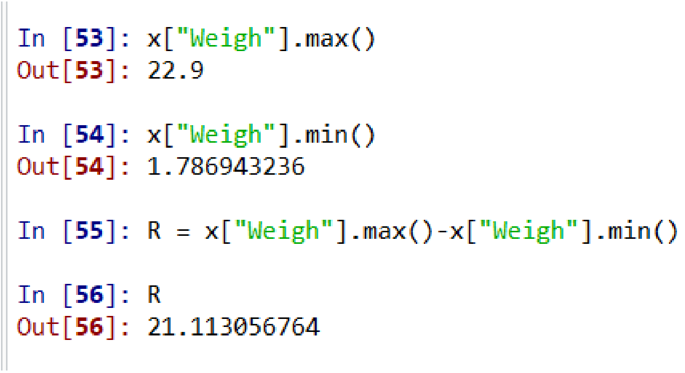
```
In [39]: x["Score"].mean()
Out[39]: 3.1258180634473685

In [40]: x["Score"].median()
Out[40]: 3.271125

In [41]: x["Score"].mode()
Out[41]:
0    3.44
Name: Score, dtype: float64

In [42]: x["Score"].std()
Out[42]: 1.0400467962343873

In [43]: x["Score"].var()
Out[43]: 1.0816973383574133
```



```
In [44]: x["Score"].max()
Out[44]: 5.424

In [45]: x["Score"].min()
Out[45]: 0.957378968

In [46]: R = x["Score"].max()-x["Score"].min()

In [47]: R
Out[47]: 4.466621032000001
```



```
In [48]: x["Weigh"].mean()
Out[48]: 16.76628577276316

In [49]: x["Weigh"].median()
Out[49]: 17.51

In [50]: x["Weigh"].mode()
Out[50]:
0    17.02
Name: Weigh, dtype: float64

In [51]: x["Weigh"].std()
Out[51]: 4.084298164222668

In [52]: x["Weigh"].var()
Out[52]: 16.681491494272656
```



```
In [53]: x["Weigh"].max()
Out[53]: 22.9

In [54]: x["Weigh"].min()
Out[54]: 1.786943236

In [55]: R = x["Weigh"].max()-x["Weigh"].min()

In [56]: R
Out[56]: 21.113056764
```

Inference:

1)The average value of the “Points” column is approximately 3.402.

“Score” column is approximately 3.125.

“Weigh” column is approximately 16.766.

It says that the average point value falls around this number.

2) The standard deviation value of the “Points” column is approximately 0.8992.

“Score” column & “Weigh” column is approximately 1.4004 & 4.0842

It measures the dispersion or spread of the data points around the mean. Simply keeping the standard deviation is the square root of the variance value.

3. The Variance value of the “Points” column is approximately 0.2858.

“Score” column is approximately 0.9573.

“Weigh” column is approximately 3.1931.

*The variance provides information about how the individual data points in a dataset deviate from the mean.

4. The Range can be calculated as the maximum value in that column – minimum value in that column.

The Range value of the “Points” column is approximately 2.17.

“Score” column is approximately 3.911.

“Weigh” column is approximately 8.4.

5.

The Median value of the “Points” column is approximately 3.695.

“Score” column is approximately 3.325.

“Weigh” column is approximately 17.71.

Here the median is the value that separates the higher half from the lower half of the data. Above are the median exact value where lowest and highest gets divided in the given dataset.

6.The mode indicates that the average value appears most frequently among the data points.

The Mode value of the “Points” column is approximately 3.695.

“Score” column is approximately 3.325.

“Weigh” column is approximately 17.71.

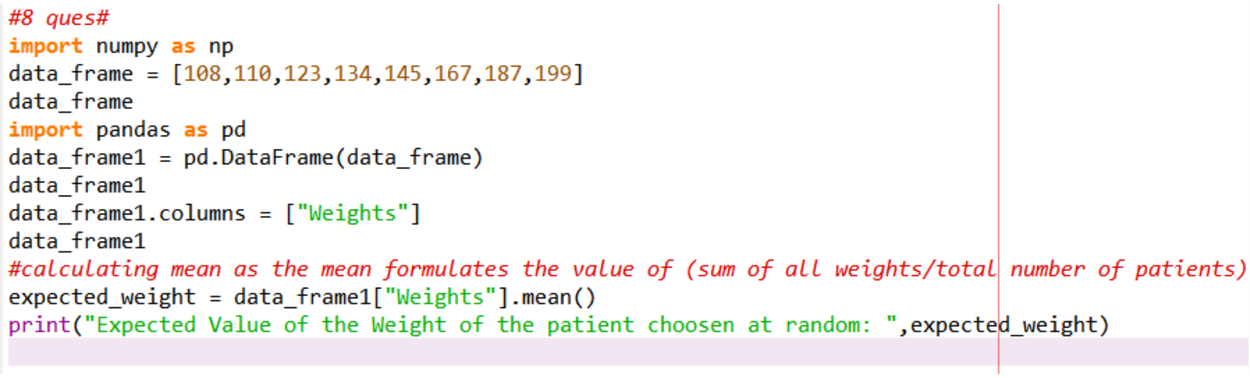
Q8) Calculate Expected Value for the problem below

a) The weights (X) of patients at a clinic (in pounds), are
108, 110, 123, 134, 135, 145, 167, 187, 199

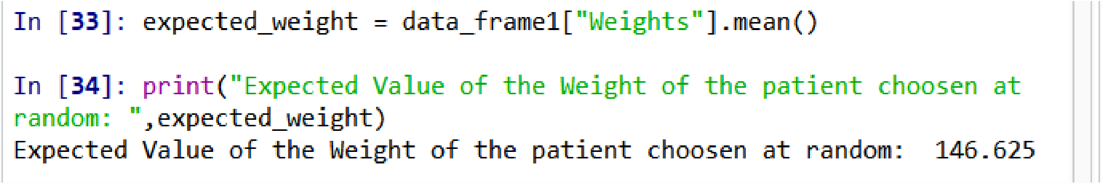
Assume one of the patients is chosen at random. What is the Expected Value of the Weight of that patient?

Expected Value = ∑Weights / Number of Patients

Total number of patients = 9 (from the above provided data)



```
#8 ques#
import numpy as np
data_frame = [108,110,123,134,145,167,187,199]
data_frame
import pandas as pd
data_frame1 = pd.DataFrame(data_frame)
data_frame1
data_frame1.columns = ["Weights"]
data_frame1
#calculating mean as the mean formulates the value of (sum of all weights/total number of patients)
expected_weight = data_frame1["Weights"].mean()
print("Expected Value of the Weight of the patient choosen at random: ",expected_weight)
```

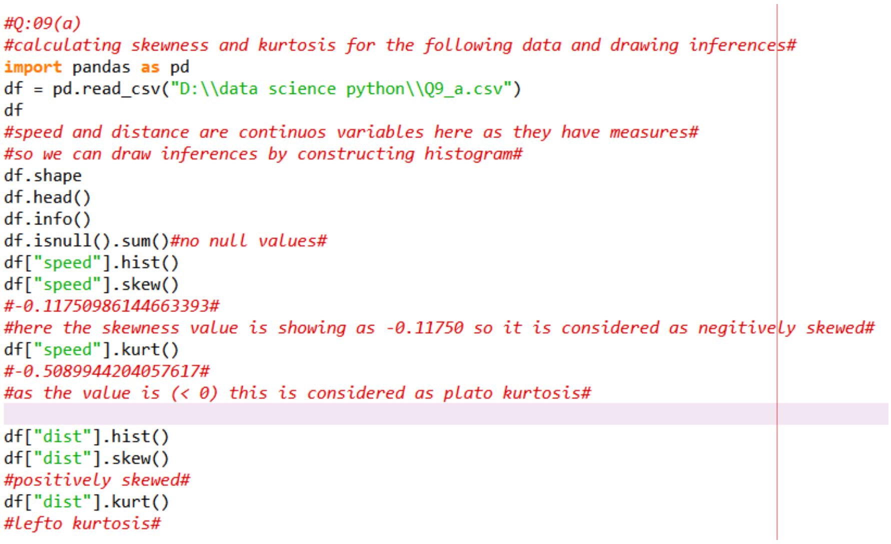


```
In [33]: expected_weight = data_frame1["Weights"].mean()

In [34]: print("Expected Value of the Weight of the patient choosen at
random: ",expected_weight)
Expected Value of the Weight of the patient choosen at random:  146.625
```

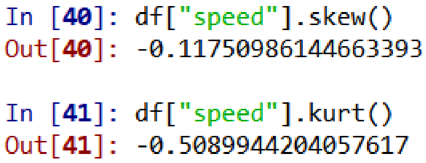
Q9) Calculate Skewness, Kurtosis & draw inferences on the following data

Cars speed and distance

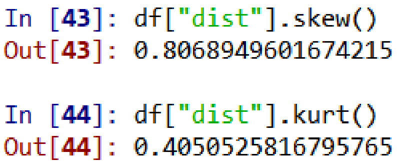
Use Q9_a.csv



```
#Q:09(a)
#calculating skewness and kurtosis for the following data and drawing inferences#
import pandas as pd
df = pd.read_csv("D:\\data science python\\Q9_a.csv")
df
#speed and distance are continuos variables here as they have measures#
#so we can draw inferences by constructing histogram#
df.shape
df.head()
df.info()
df.isnull().sum()#no null values#
df["speed"].hist()
df["speed"].skew()
#-0.11750986144663393#
#here the skewness value is showing as -0.11750 so it is considered as negitively skewed#
df["speed"].kurt()
#-0.5089944204057617#
#as the value is (< 0) this is considered as plato kurtosis#

df["dist"].hist()
df["dist"].skew()
#positively skewed#
df["dist"].kurt()
#lefto kurtosis#
```



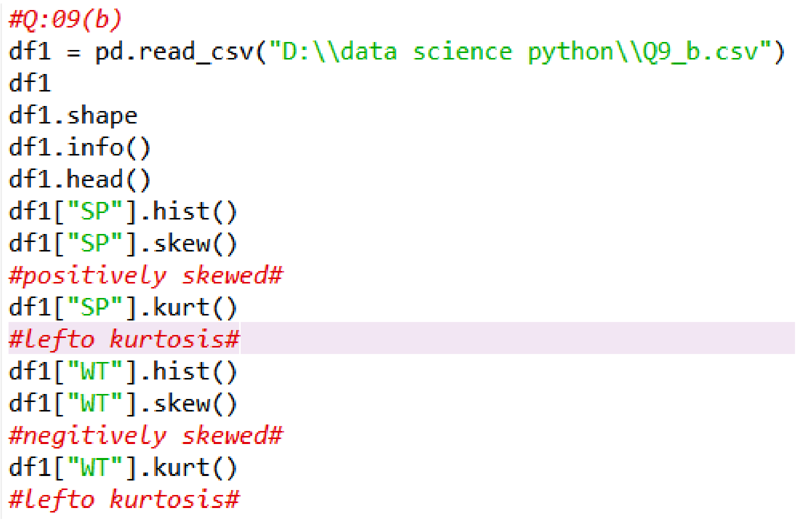
```
In [40]: df["speed"].skew()
Out[40]: -0.11750986144663393

In [41]: df["speed"].kurt()
Out[41]: -0.5089944204057617
```



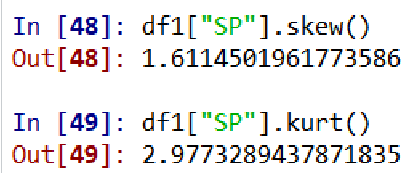
```
In [43]: df["dist"].skew()
Out[43]: 0.8068949601674215

In [44]: df["dist"].kurt()
Out[44]: 0.4050525816795765
```

SP and Weight(WT)

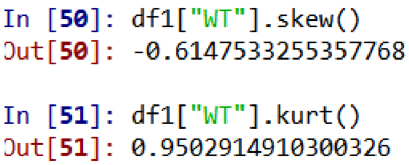
Use Q9_b.csv



```
#Q:09(b)
df1 = pd.read_csv("D:\\data science python\\Q9_b.csv")
df1
df1.shape
df1.info()
df1.head()
df1["SP"].hist()
df1["SP"].skew()
#positively skewed#
df1["SP"].kurt()
#lefto kurtosis#
df1["WT"].hist()
df1["WT"].skew()
#negitively skewed#
df1["WT"].kurt()
#lefto kurtosis#
```

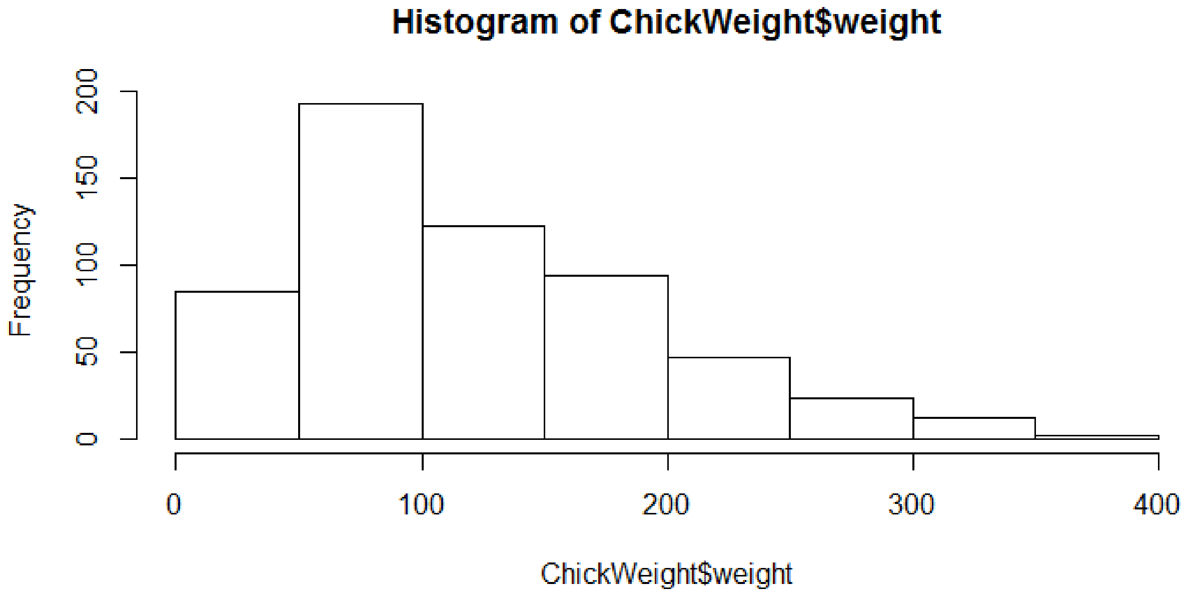


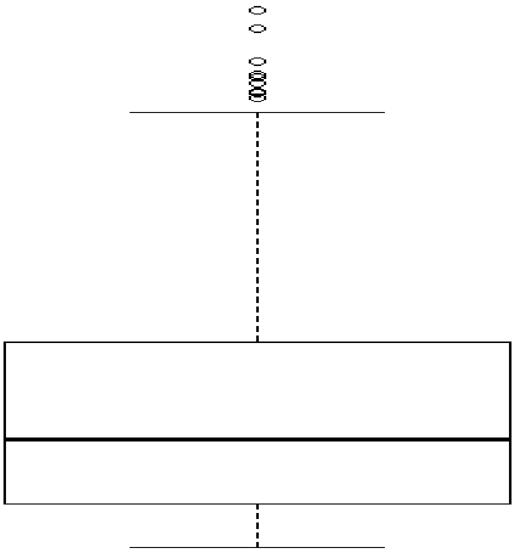
```
In [48]: df1["SP"].skew()
Out[48]: 1.6114501961773586

In [49]: df1["SP"].kurt()
Out[49]: 2.9773289437871835
```



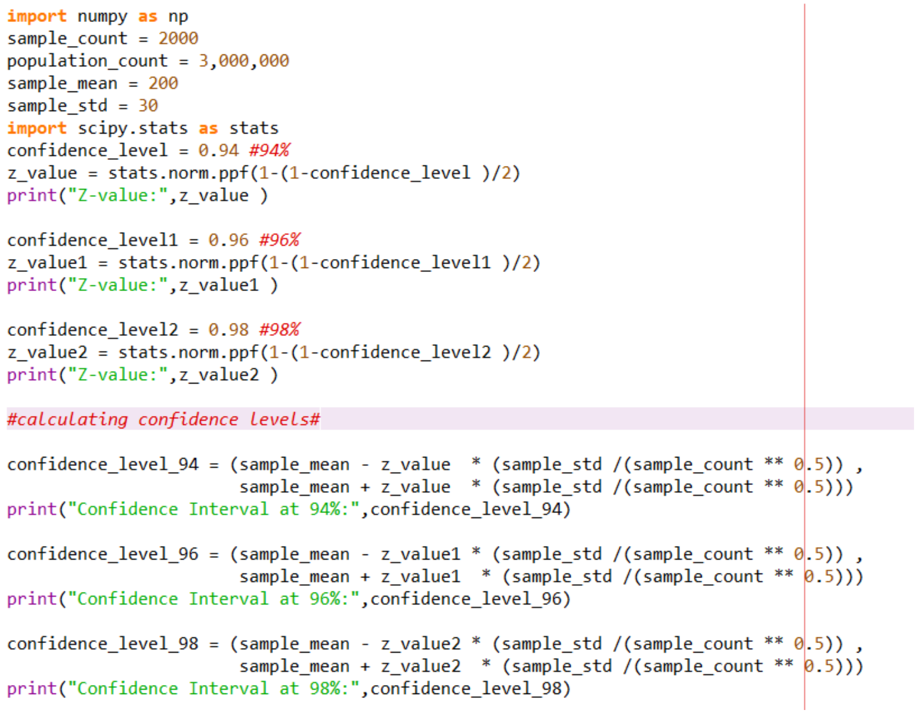
```
In [50]: df1["WT"].skew()
Out[50]: -0.6147533255357768

In [51]: df1["WT"].kurt()
Out[51]: 0.9502914910300326
```

Q10) Draw inferences about the following boxplot & histogram

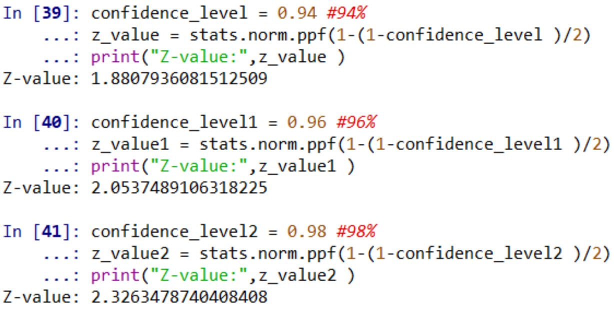




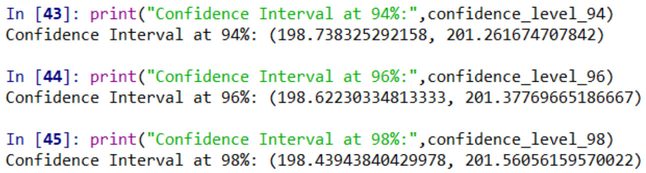
Q11) Suppose we want to estimate the average weight of an adult male in Mexico. We draw a random sample of 2,000 men from a population of 3,000,000 men and weigh them. We find that the average person in our sample weighs 200 pounds, and the standard deviation of the sample is 30 pounds. Calculate 94%,98%,96% confidence interval?



```
import numpy as np
sample_count = 2000
population_count = 3,000,000
sample_mean = 200
sample_std = 30
import scipy.stats as stats
confidence_level = 0.94 #94%
z_value = stats.norm.ppf(1-(1-confidence_level )/2)
print("Z-value:",z_value )

confidence_level1 = 0.96 #96%
z_value1 = stats.norm.ppf(1-(1-confidence_level1 )/2)
print("Z-value:",z_value1 )

confidence_level2 = 0.98 #98%
z_value2 = stats.norm.ppf(1-(1-confidence_level2 )/2)
print("Z-value:",z_value2 )

#calculating confidence levels#

confidence_level_94 = (sample_mean - z_value  * (sample_std /(sample_count ** 0.5)) ,
                       sample_mean + z_value  * (sample_std /(sample_count ** 0.5)))
print("Confidence Interval at 94%:",confidence_level_94)

confidence_level_96 = (sample_mean - z_value1 * (sample_std /(sample_count ** 0.5)) ,
                       sample_mean + z_value1  * (sample_std /(sample_count ** 0.5)))
print("Confidence Interval at 96%:",confidence_level_96)

confidence_level_98 = (sample_mean - z_value2 * (sample_std /(sample_count ** 0.5)) ,
                       sample_mean + z_value2  * (sample_std /(sample_count ** 0.5)))
print("Confidence Interval at 98%:",confidence_level_98)
```



```
In [39]: confidence_level = 0.94 #94%
    ...: z_value = stats.norm.ppf(1-(1-confidence_level )/2)
    ...: print("Z-value:",z_value )
Z-value: 1.8807936081512509

In [40]: confidence_level1 = 0.96 #96%
    ...: z_value1 = stats.norm.ppf(1-(1-confidence_level1 )/2)
    ...: print("Z-value:",z_value1 )
Z-value: 2.0537489106318225

In [41]: confidence_level2 = 0.98 #98%
    ...: z_value2 = stats.norm.ppf(1-(1-confidence_level2 )/2)
    ...: print("Z-value:",z_value2 )
Z-value: 2.3263478740408408
```



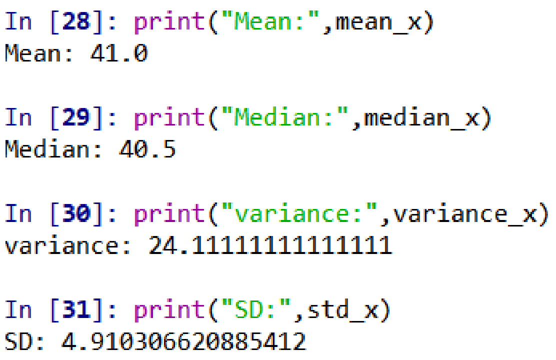
```
In [43]: print("Confidence Interval at 94%:",confidence_level_94)
Confidence Interval at 94%: (198.738325292158, 201.261674707842)

In [44]: print("Confidence Interval at 96%:",confidence_level_96)
Confidence Interval at 96%: (198.62230334813333, 201.37769665186667)

In [45]: print("Confidence Interval at 98%:",confidence_level_98)
Confidence Interval at 98%: (198.43943840429978, 201.56056159570022)
```

Q12) Below are the scores obtained by a student in tests

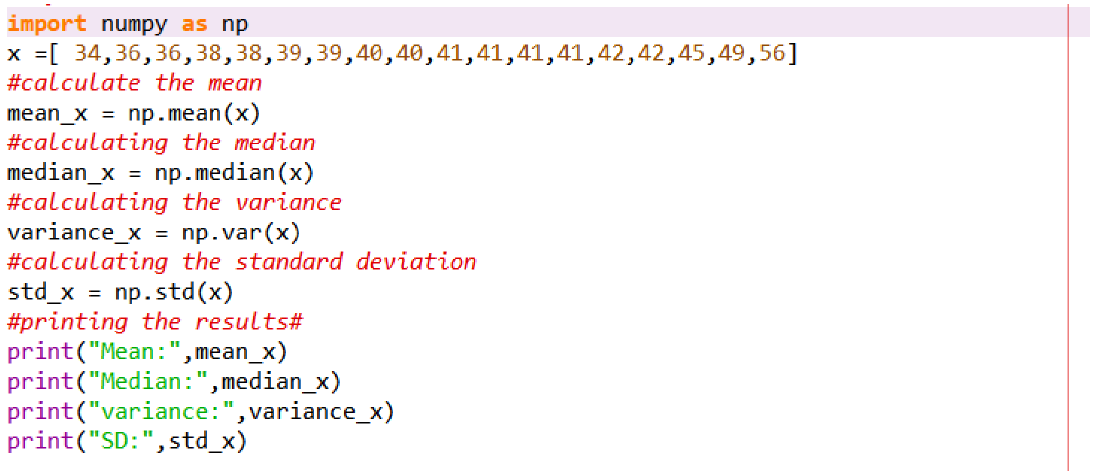
34,36,36,38,38,39,39,40,40,41,41,41,41,42,42,45,49,56

1) Find mean, median, variance, standard deviation.
2) What can we say about the student marks?

1)Solution:



```
In [28]: print("Mean:",mean_x)
Mean: 41.0

In [29]: print("Median:",median_x)
Median: 40.5

In [30]: print("variance:",variance_x)
variance: 24.11111111111111

In [31]: print("SD:",std_x)
SD: 4.910306620885412
```



```
import numpy as np
x =[ 34,36,36,38,38,39,39,40,40,41,41,41,41,42,42,45,49,56]
#calculate the mean
mean_x = np.mean(x)
#calculating the median
median_x = np.median(x)
#calculating the variance
variance_x = np.var(x)
#calculating the standard deviation
std_x = np.std(x)
#printing the results#
print("Mean:",mean_x)
print("Median:",median_x)
print("variance:",variance_x)
print("SD:",std_x)
```

2)**Mean**: The mean is the average of all the student marks. In this case, it is approximately 41.0. This indicates that, on average, the student marks are close to 41.0.

Median: The median is the middle value of the data when it is sorted in ascending order. In this case, the median is 40.5. This implies that half of the student marks are below 40.5 and other half are above 40.5.

Variance: variance measures the spread of data points from the mean. A Higher variance suggests that the data points are more spread out. In this case, the variance is approximately 24.11. The higher variance indicates that the student marks are somewhat spread out from the mean, suggesting some variability in scores.

Standard Deviation: The standard deviation is another measure of the spread of the data points from the mean .It is the square root of the variance. In this case , the standard deviation is approximately 4.91. The larger standard deviation indicates that there is a noticeable amount of variability in the student marks from the mean…

Q13) What is the nature of skewness when mean, median of data are equal?

When the mean and median of the dataset are equal, it indicates that the data is symmetrically distributed around the center. In this case , the skewness of the data is zero. Skewness is the measure of the asymmetry of the probability distribution of a real valued random variables. It tells us whether the data is skewed to the left(negative skewness) or to the right (positive skewness) relative to the mean.

Q14) What is the nature of skewness when mean > median ?

Exhibits positive skewness.

Q15) What is the nature of skewness when median > mean?

Exhibits negative skewness.

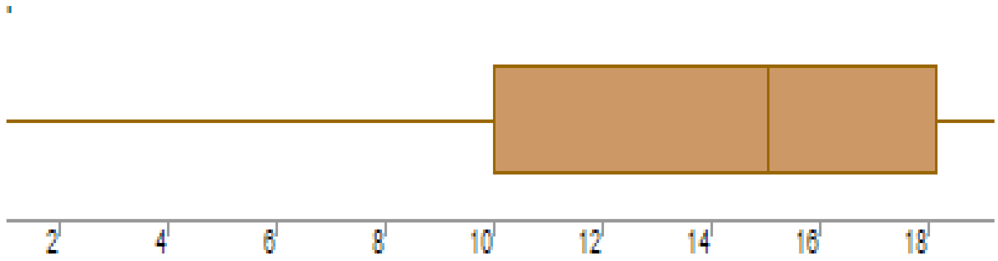
Q16) What does positive kurtosis value indicates for a data ?

Positive Kurtosis value indicates that the dataset has heavier tails or more/highly peaked or leptokurtic distribution .Positive kurtosis value always have the kurtosis value when calculated as (>0).

Q17) What does negative kurtosis value indicates for a data?

Negative Kurtosis value indicates that the dataset has lighter tails or less/flatter peaked or platykurtic distribution .Positive kurtosis value always have the kurtosis value when calculated as (<0).

Q18) Answer the below questions using the below boxplot visualization.



What can we say about the distribution of the data?

- The data is distributed across a range from around 2 to above 18.
- The upper whisker length extending beyond 18 suggests that the possibility of outliers or data points that are more spread out.
- The position of the median line between 14 and 16 suggests that the central value of the data is shifted slightly towards the higher end of the range (the line inside the boxplot indicates the median line)

- The point that the quartile range 1 being starts exactly at 10 shows that the minimum value of the data set starts from 10 and the lower 25% of data is clustered around a value around the 10.
- In the same way, the slight exceeding of Q3 beyond the 18 shows that the upper 25% of the data extends beyond 18.

What is the nature of skewness of the data?

Q2(Median):

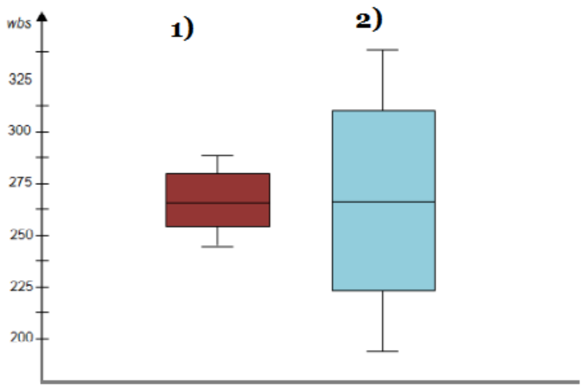
- The position Q2(median) can be calculated as the average of Q1 and Q3.
- Q1 = 10
- As Q3 exceeds 18, let's consider Q3 as 18.5
- Q2 = (Q1+Q3)/2 = (10+18.5)/2 = 28.5/2 = 14.25

In the above case, it is difficult to determine the skewness without actual distribution. However, with the information we have for now there could be a mild positive skewness. This is because the tail (right side) might be longer ,with a few higher values pulling beyond the 18.

What will be the IQR of the data (approximately)?

- The IQR is the difference between Q3 and Q1.
- Q1 = 10
- Let's assume Q3 = 18.5
- IQR = Q3 - Q1 = 18.5 – 10 = 8.5(approximately).

Q19) Comment on the below Boxplot visualizations?



Draw an Inference from the distribution of data for Boxplot 1 with respect Boxplot 2.

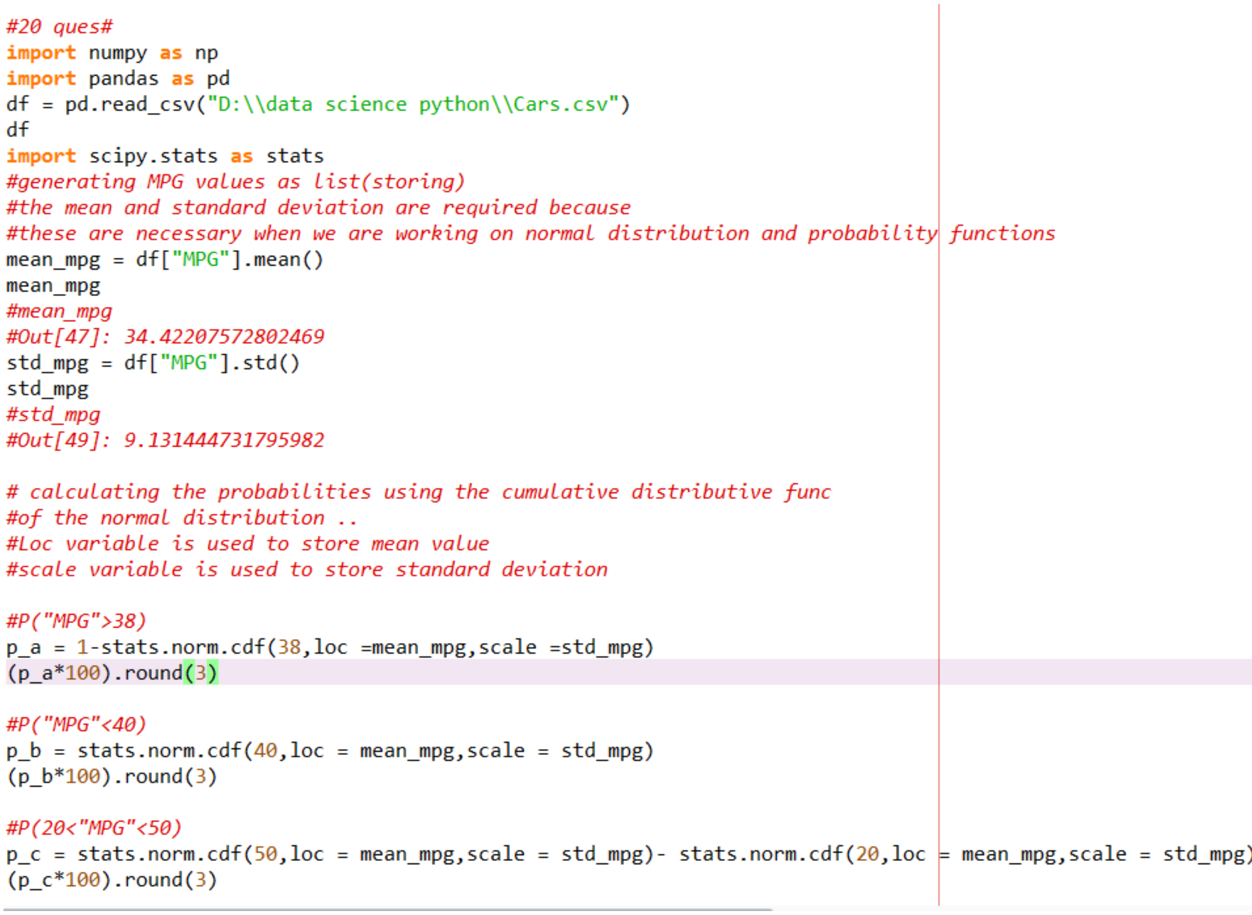
Q 20) Calculate probability from the given dataset for the below cases

Data _set: Cars.csv

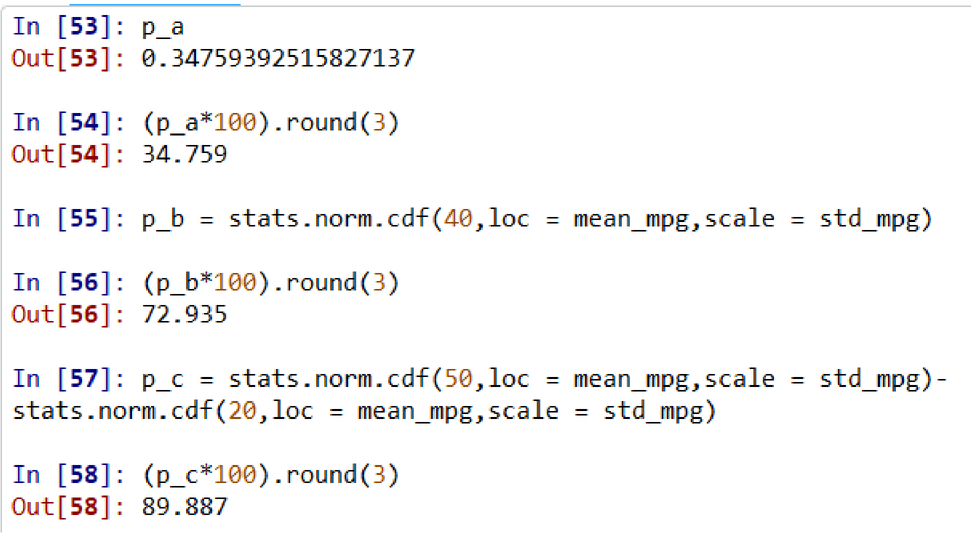
Calculate the probability of MPG of Cars for the below cases.

MPG <- Cars$MPG

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



```
#20 ques#
import numpy as np
import pandas as pd
df = pd.read_csv("D:\\data science python\\Cars.csv")
df
import scipy.stats as stats
#generating MPG values as list(storing)
#the mean and standard deviation are required because
#these are necessary when we are working on normal distribution and probability functions
mean_mpg = df["MPG"].mean()
mean_mpg
#mean_mpg
#Out[47]: 34.42207572802469
std_mpg = df["MPG"].std()
std_mpg
#std_mpg
#Out[49]: 9.131444731795982

# calculating the probabilities using the cumulative distributive func
#of the normal distribution ..
#Loc variable is used to store mean value
#scale variable is used to store standard deviation

#P("MPG">38)
p_a = 1-stats.norm.cdf(38,loc =mean_mpg,scale =std_mpg)
(p_a*100).round(3)

#P("MPG"<40)
p_b = stats.norm.cdf(40,loc = mean_mpg,scale = std_mpg)
(p_b*100).round(3)

#P(20<"MPG"<50)
p_c = stats.norm.cdf(50,loc = mean_mpg,scale = std_mpg)- stats.norm.cdf(20,loc = mean_mpg,scale = std_mpg)
(p_c*100).round(3)
```

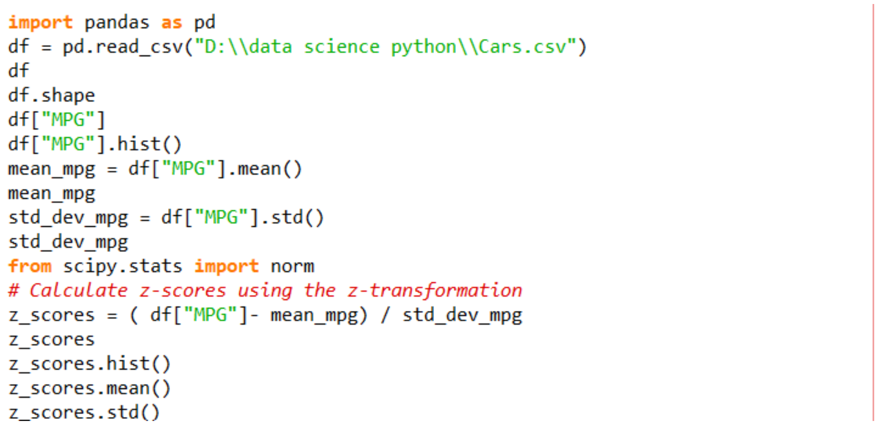


```
In [53]: p_a
Out[53]: 0.34759392515827137

In [54]: (p_a*100).round(3)
Out[54]: 34.759

In [55]: p_b = stats.norm.cdf(40,loc = mean_mpg,scale = std_mpg)

In [56]: (p_b*100).round(3)
Out[56]: 72.935

In [57]: p_c = stats.norm.cdf(50,loc = mean_mpg,scale = std_mpg)-
stats.norm.cdf(20,loc = mean_mpg,scale = std_mpg)

In [58]: (p_c*100).round(3)
Out[58]: 89.887
```

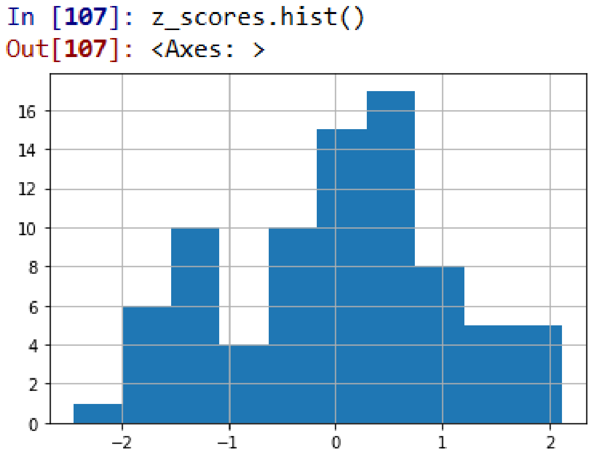
Q 21) Check whether the data follows normal distribution

a) Check whether the MPG of Cars follows Normal Distribution

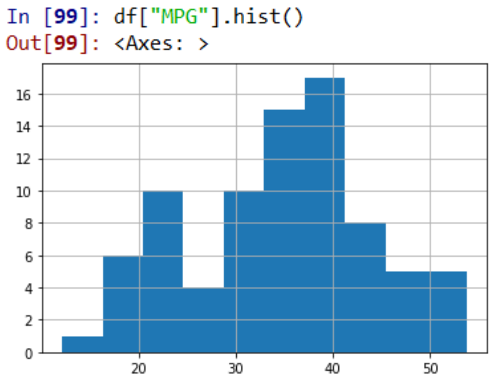
Dataset: Cars.csv



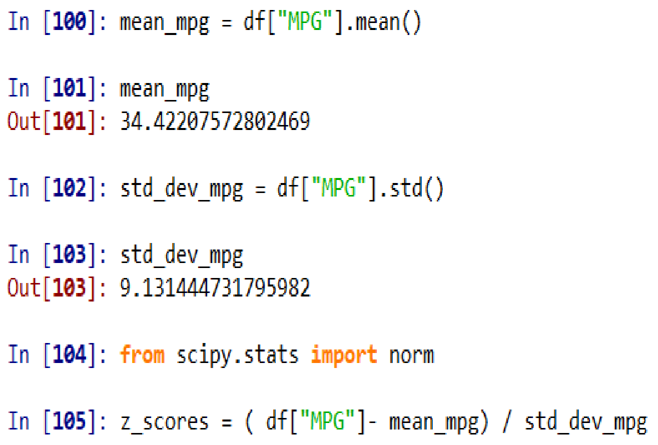
```
import pandas as pd
df = pd.read_csv("D:\\data science python\\Cars.csv")
df
df.shape
df["MPG"]
df["MPG"].hist()
mean_mpg = df["MPG"].mean()
mean_mpg
std_dev_mpg = df["MPG"].std()
std_dev_mpg
from scipy.stats import norm
# Calculate z-scores using the z-transformation
z_scores = ( df["MPG"]- mean_mpg) / std_dev_mpg
z_scores
z_scores.hist()
z_scores.mean()
z_scores.std()
```



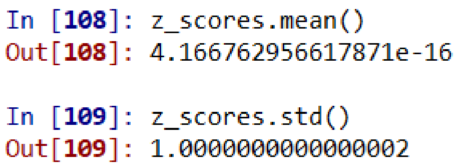
```
In [107]: z_scores.hist()
Out[107]: <Axes: >
```



```
In [99]: df["MPG"].hist()
Out[99]: <Axes: >
```



```
In [100]: mean_mpg = df["MPG"].mean()

In [101]: mean_mpg
Out[101]: 34.42207572802469

In [102]: std_dev_mpg = df["MPG"].std()

In [103]: std_dev_mpg
Out[103]: 9.131444731795982

In [104]: from scipy.stats import norm

In [105]: z_scores = ( df["MPG"]- mean_mpg) / std_dev_mpg
```

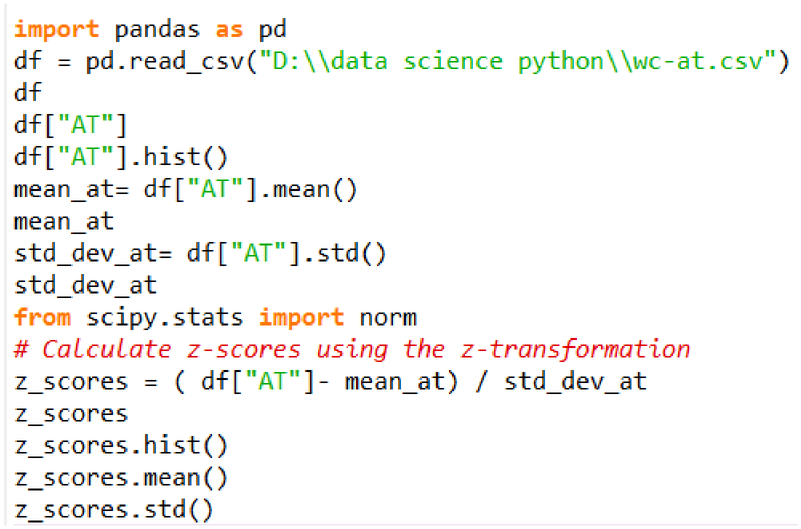


```
In [108]: z_scores.mean()
Out[108]: 4.166762956617871e-16

In [109]: z_scores.std()
Out[109]: 1.0000000000000002
```

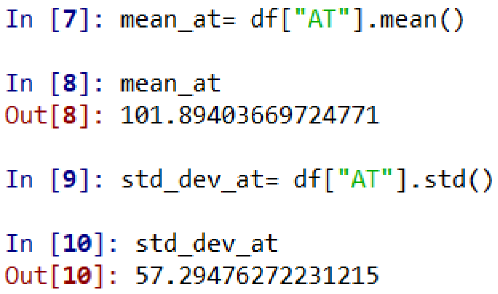
So after the Z-transformation we got the mean nearly equal to zero and standard deviation as equal to 1 so the “MPG” follows normal distribution…

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

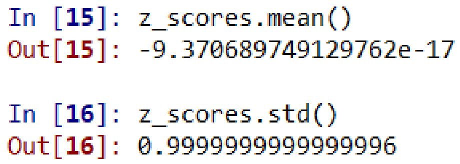
For Adipose Tissue:



```
import pandas as pd
df = pd.read_csv("D:\\data science python\\wc-at.csv")
df
df["AT"]
df["AT"].hist()
mean_at= df["AT"].mean()
mean_at
std_dev_at= df["AT"].std()
std_dev_at
from scipy.stats import norm
# Calculate z-scores using the z-transformation
z_scores = ( df["AT"]- mean_at) / std_dev_at
z_scores
z_scores.hist()
z_scores.mean()
z_scores.std()
```

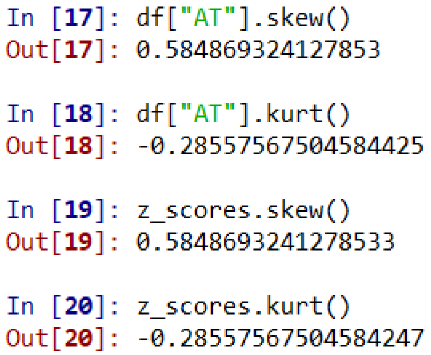


```
In [7]: mean_at= df["AT"].mean()

In [8]: mean_at
Out[8]: 101.89403669724771

In [9]: std_dev_at= df["AT"].std()

In [10]: std_dev_at
Out[10]: 57.29476272231215
```

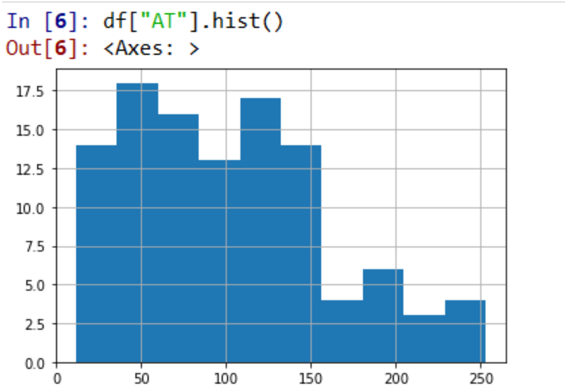


```
In [15]: z_scores.mean()
Out[15]: -9.370689749129762e-17

In [16]: z_scores.std()
Out[16]: 0.9999999999999996
```

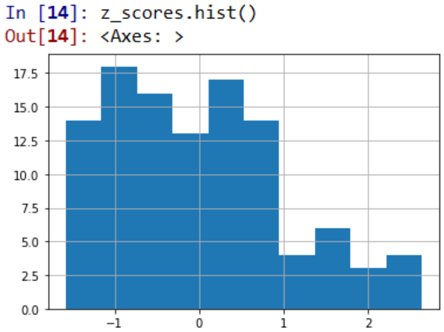
So after the z transformation we have got the values of mean and standard deviation nearly equal to 1 but not exactly equal to 1 so then we also observe skewness and kurtosis value for that variable so “For a standard normal distribution (a normal distribution with mean 0 and std 1)the skewness is 0 and also excess kurtosis (kurtosis minus 3)is also 0.but here the values for the above measures is not 0 so the above variable is said to be not following “normal distribution”.



```
In [17]: df["AT"].skew()
Out[17]: 0.584869324127853

In [18]: df["AT"].kurt()
Out[18]: -0.28557567504584425

In [19]: z_scores.skew()
Out[19]: 0.5848693241278533

In [20]: z_scores.kurt()
Out[20]: -0.28557567504584247
```



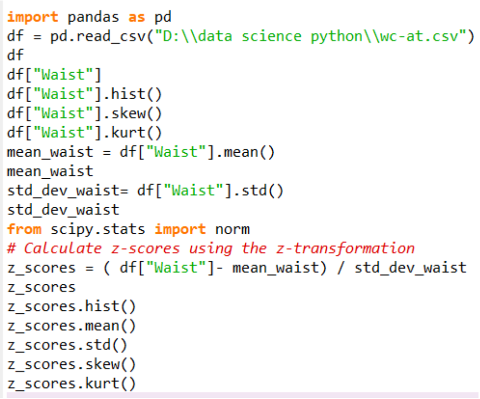
```
In [6]: df["AT"].hist()
Out[6]: <Axes: >
```



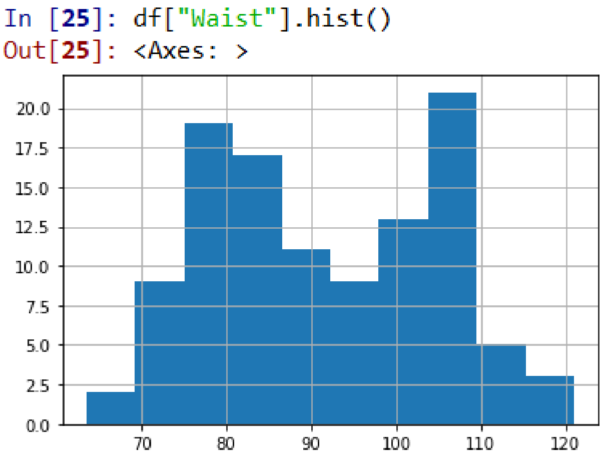
```
In [14]: z_scores.hist()
Out[14]: <Axes: >
```

The skewness and kurtosis values are only valuable insights.

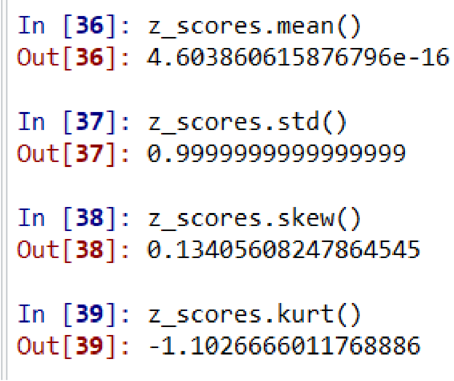
For Waist Circumference (Waist):



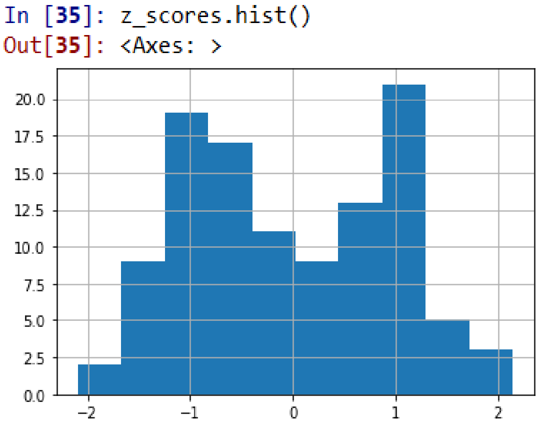
```
import pandas as pd
df = pd.read_csv("D:\\data science python\\wc-at.csv")
df
df["Waist"]
df["Waist"].hist()
df["Waist"].skew()
df["Waist"].kurt()
mean_waist = df["Waist"].mean()
mean_waist
std_dev_waist= df["Waist"].std()
std_dev_waist
from scipy.stats import norm
# Calculate z-scores using the z-transformation
z_scores = ( df["Waist"]- mean_waist) / std_dev_waist
z_scores
z_scores.hist()
z_scores.mean()
z_scores.std()
z_scores.skew()
z_scores.kurt()
```



```
In [25]: df["Waist"].hist()
Out[25]: <Axes: >
```



```
In [36]: z_scores.mean()
Out[36]: 4.603860615876796e-16

In [37]: z_scores.std()
Out[37]: 0.9999999999999999

In [38]: z_scores.skew()
Out[38]: 0.13405608247864545

In [39]: z_scores.kurt()
Out[39]: -1.1026666011768886
```



```
In [35]: z_scores.hist()
Out[35]: <Axes: >
```

As we can see after z transformation the column named “Waist” mean and standard deviation is nearly equal to zero and 1 but not exactly one so we can say this is not following the normal distribution . For our better understanding we can also take skewness and kurtosis values as powerfull and usefull insights and we are also not getting 0,1 values respectively as you can see it in the above picture.so clearly the column is not following normal distribution.

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

To calculate the Z-scores for different intervals, we can use the standard normal distribution (Z-distribution) and its percentiles. The formula to calculate the Z-scores for a specific confidence level is:

Z = $Z_{\alpha/2}$ ---→where $Z_{\alpha/2}$ is the corresponding Z-score to the desired significance level alpha.

90% Confidence Interval:

Here the confidence level is 90% ,which means alpha = 1-0.90(90%)=0.10.Half of this alpha is 0.05.when we look up the Z-score corresponding to the cumulative probability of 0.95(1-0.05) in the standard normal distribution table $Z_{0.05}$ ~1.645.

94% Confidence Interval:

Here the confidence level is 94%, which means alpha = 1-0.94(94%)=0.06.Half of this alpha is 0.03.when we look up the Z-score corresponding to the cumulative probability of 0.97(1-0.03) in the standard normal distribution table $Z_{0.05}$ ~1.880.

60% Confidence Interval:

Here the confidence level is 60%, which means alpha = 1-0.60(60%)=0.40.Half of this alpha is 0.20.when we look up the Z-score corresponding to the cumulative probability of 0.80(1-0.20) in the standard normal distribution table $Z_{0.05}$ ~1.842.

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

We will calculate the t scores when we have the sample size (<30). we would use the t-distribution instead of standard normal distribution.

$t = t_{\alpha/2,\ df}$

where $t_{\alpha/2}$ is the t-score corresponding to the desired significance level alpha.

And the degree of freedom $\mathbf{df}$ = n-1(sample size – 1).

96% Confidence Level:

The confidence level is 96% which means alpha = 1-0.96 = 0.04.so half of the value of alpha is 0.04/2 = 0.02.the degrees of freedom (df) for a sample size is 25-1 = 24.so now we have to look up the t-score corresponding to the cumulative probability of 0.980(1-0.02) and df = 24 in the t-distribution table which equals

2.398.

95% Confidence Level:

The confidence level is 95% which means alpha = 1-0.95 = 0.05.so half of the value of alpha is 0.05/2 = 0.025.the degrees of freedom (df) for a sample size is 25-1 = 24.so now we have to look up the t-score corresponding to the cumulative probability of 0.975(1-0.025) and df = 24 in the t-distribution table which equals 2.064.

99% Confidence Level:

The confidence level is 99% which means alpha = 1-0.99= 0.01.so half of the value of alpha is 0.01/2 = 0.005.the degrees of freedom (df) for a sample size is 25-1 = 24.so now we have to look up the t-score corresponding to the cumulative probability of 0.995(1-0.005) and df = 24 in the t-distribution table which equals

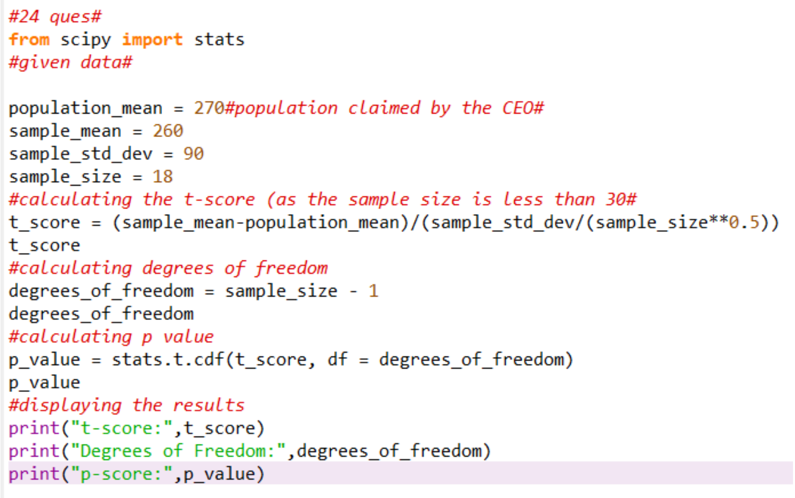
2.797.

Q 24**)** A Government company claims that an average light bulb lasts 270 days. A researcher randomly selects 18 bulbs for testing. The sampled bulbs last an average of 260 days, with a standard deviation of 90 days. If the CEO's claim were true, what is the probability that 18 randomly selected bulbs would have an average life of no more than 260 days

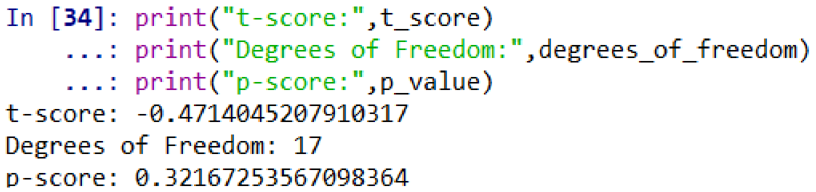
Hint:

rcode → pt(tscore,df)

df → degrees of freedom



```
#24 ques#
from scipy import stats
#given data#

population_mean = 270#population claimed by the CEO#
sample_mean = 260
sample_std_dev = 90
sample_size = 18
#calculating the t-score (as the sample size is less than 30#
t_score = (sample_mean-population_mean)/(sample_std_dev/(sample_size**0.5))
t_score
#calculating degrees of freedom
degrees_of_freedom = sample_size - 1
degrees_of_freedom
#calculating p value
p_value = stats.t.cdf(t_score, df = degrees_of_freedom)
p_value
#displaying the results
print("t-score:",t_score)
print("Degrees of Freedom:",degrees_of_freedom)
print("p-score:",p_value)
```



```
In [34]: print("t-score:",t_score)
    ...: print("Degrees of Freedom:",degrees_of_freedom)
    ...: print("p-score:",p_value)
t-score: -0.4714045207910317
Degrees of Freedom: 17
p-score: 0.32167253567098364
```

1. Formulate Hypothesis:
   NULL HYPOTHESIS (H0): the average bulb life is 270 days (CEO'S claim)
   ALTERNATIVE HYPOTHESIS (H1):
   The average bulb life is less than 270 days.
2. Alpha = 95%confidence interval I,e.,0.05.(Significance level)
3. Calculate t-scores: These measures how many standard error the sample mean is away from the population mean under the assumption of null hypothesis.

   T = sample mean-population mean/sample standard deviation /$(n)^{1/2}$

4. Find the p value and compare the p value with significance level.

Here p value is equal to 0.321 and our significance level is 0.05 so practically speaking the p value is greater than significance value then,

H0 is accepted and H1 is rejected…