Python_Assignments - Data Analytics

November 23, 2021

0.0.1 Python Assignments - Data Analytics

Assignment Question - List 1.Write a Python program which accepts a sequence of comma-separated numbers from user and generate a list and a tuple with those numbers.

[1]:
```
###### Code:
Input = input("Enter the Some numbers") #Taking User input
List = list(Input.split(","))           #spliting by commas and creating a List using list()
Tuple = tuple(Input.split(","))         #spliting by commas and creating a tuple using tuple()
#Printing the list and tuple
print("The List is:", List)
print("The Tuple is:", Tuple)
```

```
Enter the Some numbers3,5,7,23
The List is: ['3', '5', '7', '23']
The Tuple is: ('3', '5', '7', '23')
```

2.Write a Python program to display the first and last colors from the following list.

color_list = ["Red","Green","White" ,"Black"]

[2]:
```
#Defining The List
color_list = ["Red","Green","White" ,"Black"]
#Printing the Two Required Elements
print(color_list[0],color_list[3])
```

```
Red Black
```

3.Write a Python program to print the even numbers from a given list.

Sample List : [1, 2, 3, 4, 5, 6, 7, 8, 9]

[3]:
```
#Creating the List and an Empty list for Even number
List = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Even_list = []
#Looping Each Element of list
for i in List:
    if i % 2 == 0:          #Checking For Even number with a Condition
        Even_list.append(i) #Appending the even_list with even number
```

```
print(Even_list)
```

```
[2, 4, 6, 8]
```

Assignment Question - Module 1.Write a Python program to calculate number of days between two dates. Hint: use Datetime package/module.

[4]:
```
#Code:
#importing Datetime module
import datetime as dt
First_Date = dt.date(2014, 7, 2)
Last_Date = dt.date(2014, 7, 11)
diff = Last_Date - First_Date
print("The Difference Between Two Given Date is",diff)
```

```
The Difference Between Two Given Date is 9 days, 0:00:00
```

Assignment Question - Function 1.Write a Python program to get the volume of a sphere with radius 6.

[5]:
```
#Volume of Sphere = 4/3 pi* r**3
pi = 3.14
#Defining a Function For Volume of Sphere with one Parameter of Radius
def vfs(r):
    v = 4 / 3 * pi * r**3
    print("When Radius is", r,"then Volume of Sphere:",v)
#Calling the Function Defined - with Argument as Radius 6
vfs(6)
```

```
When Radius is 6 then Volume of Sphere: 904.3199999999999
```

2.Write a Python program to calculate the sum of three given numbers, if the values are equal then return three times of their sum hint: write User defined functions

[6]:
```
#Code:
#Defining the Function
def sumequal(num1,num2,num3):
    if num1==num2==num3:
        snum = (num1 + num2 + num3) * 3
        print("three times of their sum",snum)
    else:
        print("Number are Not Equal")

#Calling the Function
sumequal(10,10,10)
```

```
three times of their sum 90
```

3.Write a Python program to count the number 4 in a given list.

List = [1,4,6,8,4,9,4]

[7]:

```
#Code
List = [1,4,6,8,4,9,4]
print("The Count of Number 4 is:",List.count(4))
```

```
The Count of Number 4 is: 3
```

4.Write a Python program to print all even numbers from a given numbers list in the same order and stop the printing if any numbers that come after 237 in the sequence.

List = [399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687, 217,

815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742, 717,

958,743, 527]

[8]:

```
#Code
List = [399,162,758,219,918,237,412,566,826,248,866,950,626,949,687,217,
       ␣
 ↪815,67,104,58,512,24,892,894,767,553,81,379,843,831,445,742,717,958,743,527]
for i in List:
    if i == 237:
         break    #Using Break Statement to break loop when 237 comes
    if i %2 == 0:
      print(i)
```

```
162
758
918
```

5.Write a Python program to find those numbers which are divisible by 7 and multiple of 5, between 1500 and 2700 (both included)

[9]:

```
#Code
nlist = []
#Defining the Range First
rng = range(1500,2701)
for i in rng:
    if i%7 == 0 and i%5 == 0:   #Checking the Condition Using if Statements
        nlist.append(i)
#Printing the list of Number
print(nlist)
```

```
[1505, 1540, 1575, 1610, 1645, 1680, 1715, 1750, 1785, 1820, 1855, 1890, 1925,
1960, 1995, 2030, 2065, 2100, 2135, 2170, 2205, 2240, 2275, 2310, 2345, 2380,
2415, 2450, 2485, 2520, 2555, 2590, 2625, 2660, 2695]
```

6.Write a Python program that prints all the numbers from 0 to 6 except 3 and 6.

[10]:

```
#Code
#Defining the Range First
rng = range(7)
for i in rng:
    if i == 3 or i == 6:  #Checking the Condition Using if Statements
        continue
    print(i,end=" ")
```

```
0 1 2 4 5
```

7.Write a Python program to get the Fibonacci series between 0 to 50.

[11]:

```
#Code
a = 0
b = 1
while b < 50:
    print(b)
    a, b = b, a + b
```

```
1
1
2
3
5
8
13
21
34
```

8.Write a Python program to get the Fibonacci series between 0 to 50.

[12]:

```
#Code
a = 0
b = 1
while b < 50:
    print(b)
    a, b = b, a + b
```

```
1
1
2
3
5
8
13
21
34
```

9.Write a Python function that takes a list and returns a new list with unique elements of the first list.

[13]:

```
#Code
Uni = []
def unique_list(List):
```

```python
    for i in List:
        if i not in Uni:
            Uni.append(i)
    print("The Unique List is",Uni)

#Calling The Function
unique_list([1,2,3,3,3,3,4,5])
```

```
The Unique List is [1, 2, 3, 4, 5]
```

Assignment Question - String 1.Write a Python program to concatenate all elements in a list into a string and return it.

[14]:

```python
List = ['A','k','a','a','s','h']
string =''
for i in List:
    string += str(i)
print("The String is:",string)
```

```
The String is: Akaash
```

Assignment Question - Dictionary 1.Write a Python script to concatenate following dictionaries to create a new one.

[15]:

```python
dic1={1:10, 2:20}
dic2={3:30, 4:40}
dic3={5:50,6:60}
Dict_Final = {}
#concatenate / Mergeing the Dictionaries
Dict_Final.update(dic1)      #Merging First Dictionary
Dict_Final.update(dic2)      #Merging second Dictionary with First
Dict_Final.update(dic3)     #Merging Third Dictionary with First & Second
#Printing the Final Dictionary
print(Dict_Final)
```

```
{1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
```

Assignment Question - Series 1.Write a Python program to add, subtract, multiple and divide two Pandas Series.

Sample Series: [2, 4, 6, 8, 10], [1, 3, 5, 7, 9]

[16]:

```python
#Code
#importing Library
import pandas as pd
#Creating the Two Series
S1 = pd.Series([2, 4, 6, 8, 10])
S2 = pd.Series([1, 3, 5, 7, 9])
#Adding Two Series
```

```
Add = S1 + S2
print("Adding is:")
print(Add)
#substract Two Series
Sub = S1 - S2
print("Substract is:")
print(Sub)
#Multiply Two Series
Mul = S1 * S2
print("Multiple is:")
print(Mul)
#Divide Two Series
Div = S1 * S2
print("Divide is:")
print(Div)
```

```
Adding is:
0     3
1     7
2    11
3    15
4    19
dtype: int64
Substract is:
0    1
1    1
2    1
3    1
4    1
dtype: int64
Multiple is:
0     2
1    12
2    30
3    56
4    90
dtype: int64
Divide is:
0     2
1    12
2    30
3    56
4    90
dtype: int64
```

Assignment Question - DataFrame 1.Write a Pandas program to select the specified columns and rows from a given data frame. And Select the Specific Column Given in the Problem Statement on LMS

[17]:

```
import pandas as pd
import numpy as np

Data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael',␣
 ↪'Matthew', 'Laura', 'Kevin', 'Jonas'],
        'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',␣
 ↪'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(Data , index=labels)
print(df)
print("\n") # Line Break
#Selecting The Expected Output
print("Select specific columns and rows:")
print(df.iloc[[1, 3, 5, 6], [1, 3]])
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes


Select specific columns and rows:
   score qualify
b    9.0      no
d    NaN      no
f   20.0     yes
g   14.5     yes
```

2.Use Crime dataset from LMS

I) find the aggregations like all moments of business decisions for all columns,value counts.

II) do the plottings like plottings like histogram, boxplot, scatterplot, barplot, piechart,dot chart.

[18]:

```
#Code - find the aggregations like all moments of business decisions for all␣
 ↪columns,value counts.

#importing The Required Library
import pandas as pd
```

```python
Dataset = pd.read_csv("C:/Users/Akaash/Downloads/crime_data.csv")
#aggregations
print(Dataset.describe())
#Value_Count() - Murders
Dataset.value_counts('Murder')
```

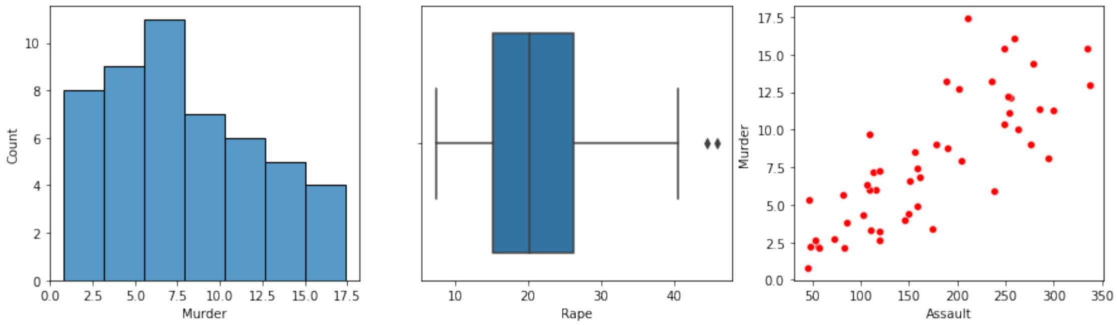
```
         Murder    Assault   UrbanPop       Rape
count  50.00000  50.000000  50.000000  50.000000
mean    7.78800 170.760000  65.540000  21.232000
std     4.35551  83.337661  14.474763   9.366385
min     0.80000  45.000000  32.000000   7.300000
25%     4.07500 109.000000  54.500000  15.075000
50%     7.25000 159.000000  66.000000  20.100000
75%    11.25000 249.000000  77.750000  26.175000
max    17.40000 337.000000  91.000000  46.000000
```

```
[18]: Murder
      2.2     2
      2.6     2
      2.1     2
      15.4    2
      13.2    2
      9.0     2
      6.0     2
      0.8     1
      11.1    1
      8.8     1
      9.7     1
      10.0    1
      10.4    1
      12.1    1
      11.3    1
      11.4    1
      8.1     1
      12.2    1
      12.7    1
      13.0    1
      14.4    1
      16.1    1
      8.5     1
      7.3     1
      7.9     1
      4.4     1
      2.7     1
      3.2     1
      3.3     1
      3.4     1
```

```
3.8     1
4.0     1
4.3     1
4.9     1
7.4     1
5.3     1
5.7     1
5.9     1
6.3     1
6.6     1
6.8     1
7.2     1
17.4    1
dtype: int64
```

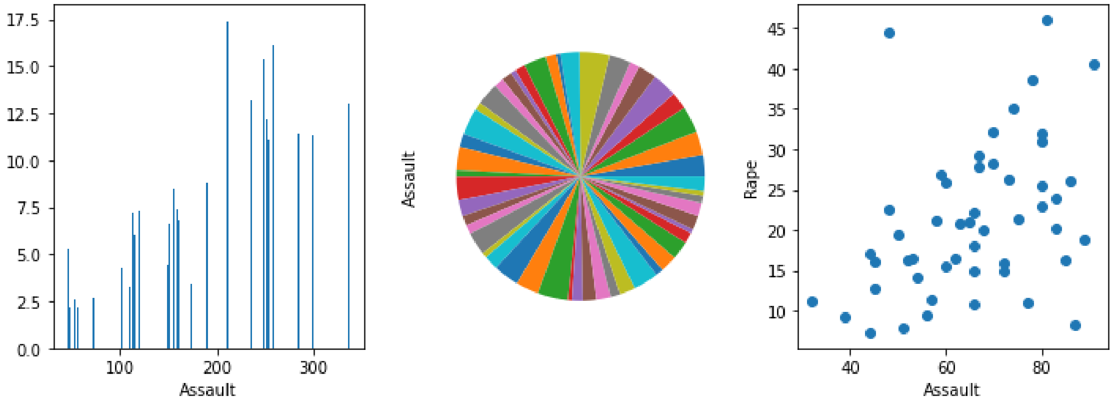
[19]:

```
#Code II)do the plottings like plottings like histogram, boxplot, scatterplot,␣
 ↪barplot, piechart,dot chart.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

import warnings
warnings.filterwarnings("ignore")  #--to ignore warnings

Dataset = pd.read_csv("C:/Users/Akaash/Downloads/crime_data.csv")

#Histogram
plt.subplots(figsize = (15,4))
plt.subplot(1,3,1)
sns.histplot(x = "Murder",data = Dataset)
#Boxplot
plt.subplot(1,3,2)
sns.boxplot(Dataset['Rape'], orient = "v")
#Scatterplot
plt.subplot(1,3,3)
sns.scatterplot(x = "Assault", y = "Murder", data = Dataset, color ='red')
plt.show()
```



[20]:

```
#piechart
plt.subplots(figsize = (12,4))
plt.subplot(1,3,1)
plt.bar("Assault","Murder",data = Dataset)
plt.xlabel('Assault')
plt.subplot(1,3,2)
plt.pie("Assault", data = Dataset)
plt.ylabel('Assault')
plt.subplot(1,3,3)
plt.scatter("UrbanPop","Rape", data = Dataset)
plt.xlabel("Assault")
plt.ylabel("Rape")
plt.show()
```



3.use mtcars dataset from LMS

A) delete/ drop rows-10 to 15 of all columns

B)drop the VOL column

C)write the forloop to get value_counts of all cloumns

[21]:

```
#importing The Required Library
import pandas as pd
Dataset = pd.read_csv("C:/Users/Akaash/Downloads/mtcars.csv")
#A) delete/ drop rows-10 to 15 of all columns
Dataset.drop([10,11,12,13,14,15],axis=0,inplace=True) #axis=0 preform on rows,

#B)drop the VOL column
Dataset.drop(["vs"], axis = 1, inplace = True)

#Dataset after Dropping and row and Column
Dataset
```

[21]:

```
     mpg  cyl   disp   hp  drat     wt   qsec  am  gear  carb
0   21.0    6  160.0  110  3.90  2.620  16.46   1     4     4
1   21.0    6  160.0  110  3.90  2.875  17.02   1     4     4
2   22.8    4  108.0   93  3.85  2.320  18.61   1     4     1
3   21.4    6  258.0  110  3.08  3.215  19.44   0     3     1
4   18.7    8  360.0  175  3.15  3.440  17.02   0     3     2
5   18.1    6  225.0  105  2.76  3.460  20.22   0     3     1
6   14.3    8  360.0  245  3.21  3.570  15.84   0     3     4
7   24.4    4  146.7   62  3.69  3.190  20.00   0     4     2
8   22.8    4  140.8   95  3.92  3.150  22.90   0     4     2
9   19.2    6  167.6  123  3.92  3.440  18.30   0     4     4
16  14.7    8  440.0  230  3.23  5.345  17.42   0     3     4
17  32.4    4   78.7   66  4.08  2.200  19.47   1     4     1
18  30.4    4   75.7   52  4.93  1.615  18.52   1     4     2
19  33.9    4   71.1   65  4.22  1.835  19.90   1     4     1
20  21.5    4  120.1   97  3.70  2.465  20.01   0     3     1
21  15.5    8  318.0  150  2.76  3.520  16.87   0     3     2
22  15.2    8  304.0  150  3.15  3.435  17.30   0     3     2
23  13.3    8  350.0  245  3.73  3.840  15.41   0     3     4
24  19.2    8  400.0  175  3.08  3.845  17.05   0     3     2
25  27.3    4   79.0   66  4.08  1.935  18.90   1     4     1
26  26.0    4  120.3   91  4.43  2.140  16.70   1     5     2
27  30.4    4   95.1  113  3.77  1.513  16.90   1     5     2
28  15.8    8  351.0  264  4.22  3.170  14.50   1     5     4
29  19.7    6  145.0  175  3.62  2.770  15.50   1     5     6
30  15.0    8  301.0  335  3.54  3.570  14.60   1     5     8
31  21.4    4  121.0  109  4.11  2.780  18.60   1     4     2
```

[22]:

```
#C)write the forloop to get value_counts of all cloumns
for c in Dataset.columns:
    print( Dataset[c].value_counts())
```

```
21.0    2
19.2    2
30.4    2
22.8    2
```

```
21.4    2
18.7    1
21.5    1
15.2    1
27.3    1
15.0    1
15.5    1
19.7    1
26.0    1
15.8    1
24.4    1
13.3    1
18.1    1
33.9    1
14.3    1
14.7    1
32.4    1
Name: mpg, dtype: int64
4    11
8     9
6     6
Name: cyl, dtype: int64
160.0    2
360.0    2
78.7     1
440.0    1
350.0    1
95.1     1
140.8    1
120.3    1
120.1    1
318.0    1
351.0    1
301.0    1
108.0    1
75.7     1
225.0    1
145.0    1
71.1     1
121.0    1
304.0    1
167.6    1
258.0    1
79.0     1
146.7    1
400.0    1
Name: disp, dtype: int64
110    3
```

```
175    3
150    2
245    2
66     2
65     1
335    1
62     1
93     1
123    1
52     1
91     1
113    1
109    1
105    1
264    1
97     1
230    1
95     1
Name: hp, dtype: int64
3.92    2
3.90    2
3.15    2
3.08    2
4.08    2
4.22    2
2.76    2
3.70    1
4.43    1
3.62    1
3.69    1
4.93    1
3.23    1
4.11    1
3.77    1
3.85    1
3.54    1
3.21    1
3.73    1
Name: drat, dtype: int64
3.570    2
3.440    2
2.200    1
2.780    1
3.845    1
3.170    1
1.835    1
2.320    1
2.140    1
```

```
3.460    1
2.770    1
3.215    1
2.465    1
1.513    1
3.840    1
2.620    1
3.435    1
3.190    1
5.345    1
3.150    1
2.875    1
1.615    1
1.935    1
3.520    1
Name: wt, dtype: int64
17.02    2
17.30    1
18.61    1
20.01    1
16.90    1
19.47    1
16.87    1
18.60    1
14.50    1
15.84    1
19.90    1
20.22    1
16.70    1
15.50    1
19.44    1
18.30    1
17.42    1
15.41    1
16.46    1
18.90    1
17.05    1
22.90    1
14.60    1
20.00    1
18.52    1
Name: qsec, dtype: int64
0    13
1    13
Name: am, dtype: int64
4    11
3    10
5     5
```

```
Name: gear, dtype: int64
2    10
1     7
4     7
6     1
8     1
Name: carb, dtype: int64
```

4.Use Bank Dataset from LMS

A)change all the categorical columns into numerical by creating Dummies and using label encoder.

B) rename all the column names DF

C) Rename only one specific column in DF

```
[23]: #A)change all the categorical columns into numerical by creating Dummies and␣
       ↪using label encoder.
      #importing The Required Library
      import pandas as pd
      Dataset = pd.read_csv("C:/Users/Akaash/Downloads/bank-full.csv",delimiter = ';')
      Dataset.head()
```

```
[23]:    age           job  marital  education default  balance housing loan  \
      0   58    management  married   tertiary      no     2143     yes   no
      1   44    technician   single  secondary      no       29     yes   no
      2   33  entrepreneur  married  secondary      no        2     yes  yes
      3   47   blue-collar  married    unknown      no     1506     yes   no
      4   33       unknown   single    unknown      no        1      no   no

         contact  day month  duration  campaign  pdays  previous poutcome   y
      0  unknown    5   may       261         1     -1         0  unknown  no
      1  unknown    5   may       151         1     -1         0  unknown  no
      2  unknown    5   may        76         1     -1         0  unknown  no
      3  unknown    5   may        92         1     -1         0  unknown  no
      4  unknown    5   may       198         1     -1         0  unknown  no
```

```
[24]: #A)change all the categorical columns into numerical by creating Dummies and␣
       ↪using label encoder.

      #Dummies
      #importing the Required library
      import pandas as pd
      Dataset2 = pd.get_dummies(Dataset)
      Dataset2.head()
```

```
[24]:    age  balance  day  duration  campaign  pdays  previous  job_admin.  \
      0   58     2143    5       261         1     -1         0           0
      1   44       29    5       151         1     -1         0           0
```

```
2  33      2  5     76        1    -1        0          0
3  47   1506  5     92        1    -1        0          0
4  33      1  5    198        1    -1        0          0

   job_blue-collar  job_entrepreneur  …  month_may  month_nov  month_oct  \
0                0                 0  …          1          0          0
1                0                 0  …          1          0          0
2                0                 1  …          1          0          0
3                1                 0  …          1          0          0
4                0                 0  …          1          0          0

   month_sep  poutcome_failure  poutcome_other  poutcome_success  \
0          0                 0               0                 0
1          0                 0               0                 0
2          0                 0               0                 0
3          0                 0               0                 0
4          0                 0               0                 0

   poutcome_unknown  y_no  y_yes
0                 1     1      0
1                 1     1      0
2                 1     1      0
3                 1     1      0
4                 1     1      0

[5 rows x 53 columns]
```

Inference: One Hot Encoding is Applied

[25]:

```python
#A)change all the categorical columns into numerical by creating Dummies and␣
 ↪using label encoder.

#Label Encoder
#note: Label Encoder Are used only when Y variable is Categorial
#importing the Required library
from sklearn.preprocessing import LabelEncoder

# Completing the dataset - applying lable encoder to Catergorial column
Dataset3 = Dataset.apply(LabelEncoder().fit_transform)
Dataset3.head()
```

[25]:

```
   age  job  marital  education  default  balance  housing  loan  contact  \
0   40    4        1          2        0     3036        1     0        2
1   26    9        2          1        0      945        1     0        2
2   15    2        1          1        0      918        1     1        2
3   29    1        1          3        0     2420        1     0        2
4   15   11        2          3        0      917        0     0        2
```

```
   day  month  duration  campaign  pdays  previous  poutcome  y
0    4      8       261         0      0         0         3  0
1    4      8       151         0      0         0         3  0
2    4      8        76         0      0         0         3  0
3    4      8        92         0      0         0         3  0
4    4      8       198         0      0         0         3  0
```

Inference: Label Encoder is Applied

[26]:
```
#B)rename all the column names DF
#rename the Solar column axis=1 perform it based on column
Dataset4 = Dataset3.rename({'age': 'AGE','job':'JOB','marital':
 ↪'MARITAL','education':'EDUCATION','default':'DEFAULT',
                           'balance':'BALANCE','housing':'HOUSING','loan':
 ↪'LOAN','contact':'CONTACT','day':'DAY',
                           'month':'MONTH','Duration':'DURATION','campaign':
 ↪'CAMPAIGN','pdays':'PDAYS',
                           'previous':'PREVIOUS','poutcome':'POUTCOME','y':
 ↪'Y'}, axis=1)
print(Dataset4.columns)
```

```
Index(['AGE', 'JOB', 'MARITAL', 'EDUCATION', 'DEFAULT', 'BALANCE', 'HOUSING',
       'LOAN', 'CONTACT', 'DAY', 'MONTH', 'duration', 'CAMPAIGN', 'PDAYS',
       'PREVIOUS', 'POUTCOME', 'Y'],
      dtype='object')
```

Inference: Renamed Using rename()

[27]:
```
#C) Rename only one specific column in DF
#rename the Solar column axis=1 perform it based on column
Dataset5 = Dataset4.rename({'AGE': 'CUS_AGE'}, axis=1)
print(Dataset5.columns)
```

```
Index(['CUS_AGE', 'JOB', 'MARITAL', 'EDUCATION', 'DEFAULT', 'BALANCE',
       'HOUSING', 'LOAN', 'CONTACT', 'DAY', 'MONTH', 'duration', 'CAMPAIGN',
       'PDAYS', 'PREVIOUS', 'POUTCOME', 'Y'],
      dtype='object')
```

Inference: AGE Column Name renamed to CUS_AGE

[28]:
```
#5. After doing all the changes in bank data(Q19). save the file in your␣
 ↪directory in Csv Format.

#Saving the DataFrame into csv
Dataset5.to_csv("C:/Users/Akaash/Downloads/Q5_Updated.csv")

Q5_Updated = pd.read_csv("C:/Users/Akaash/Downloads/Q5_Updated.csv")
Q5_Updated.head()
```

[28]:

```
   Unnamed: 0  CUS_AGE  JOB  MARITAL  EDUCATION  DEFAULT  BALANCE  HOUSING  \
0           0       40    4        1          2        0     3036        1
1           1       26    9        2          1        0      945        1
2           2       15    2        1          1        0      918        1
3           3       29    1        1          3        0     2420        1
4           4       15   11        2          3        0      917        0

   LOAN  CONTACT  DAY  MONTH  duration  CAMPAIGN  PDAYS  PREVIOUS  POUTCOME  Y
0     0        2    4      8       261         0      0         0         3  0
1     0        2    4      8       151         0      0         0         3  0
2     1        2    4      8        76         0      0         0         3  0
3     0        2    4      8        92         0      0         0         3  0
4     0        2    4      8       198         0      0         0         3  0
```

Inference: Changed made Dataset is saved as csv and import the same

Assignment Question - Basic Program 1.Write Python Programs to use various operators in Python

[29]:

```python
a = 9
b = 4
# Addition of numbers
add = a + b
# Subtraction of numbers
sub = a - b
# Multiplication of number
mul = a * b
# Division(float) of number
div1 = a / b
# Division(floor) of number
div2 = a // b
# Modulo of both number
mod = a % b
# Power
p = a ** b
# print results
print("Examples of Arithmetic Operator")
print(add)
print(sub)
print(mul)
print(div1)
print(div2)
print(mod)
print(p)

print("Comparsion operator using if statement")

if a < b:
    print("b is greater")
elif a > b:
    print("a is Greater")
else:
    print("a & b is equal")
```

```
Examples of Arithmetic Operator
13
5
36
2.25
2
1
6561
Comparsion operator using if statement
a is Greater
```

2.Create list of elements and slice and dice it

[30]:

```
# Initialize list
List = [1, 2, 3, 4, 5, 6, 7, 8, 9]
# Show original list
print("\nOriginal List:\n", List)
print("\nSliced Lists: ")
# Display sliced list
print(List[3:9:2])
# Display sliced list
print(List[::2])
# Display sliced list
print(List[::])
```

```

Original List:
 [1, 2, 3, 4, 5, 6, 7, 8, 9]

Sliced Lists:
[4, 6, 8]
[1, 3, 5, 7, 9]
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

3.Using while loop accept numbers until sum of numbers is less than 100

[31]:

```
#Code
num = 1
while num<100: #It will run till 99
    print(num)
    num = num+1 #will Add the num +1 at every iteration of the loop
else:
    print("Num is now greater then 100")
```

```
97
98
99
Num is now greater then 100
```

4.Write a python program Read & write Excel files

[32]:

```
#Loading the Pandas Library
import pandas as pd
#Reading the Excel into Dataframe
df = pd.read_excel('C:/Users/Akaash/Downloads/Q4.xlsx')
#Printing the Excel Dataframe
print(df)
# for One Line Space
print('\n')
#Printing the Shape of Excel Dataframe
print(df.shape)
#Creating a DataFrame to Add to Excel Dataframe
write_df = pd.DataFrame({'cars':['Cadillac Fleetwood','Lincoln Continental'],
                          'Points':[2.93,3],'Score':[5.25,5.42],'Weigh':[17.
 ↪98,17.82]})
#Printing the New Dataframe
print(write_df)
# for One Line Space
print('\n')
#Concatnating the Dataframe and Resetting the Index
written = pd.concat([df, write_df]).reset_index()
#Dropping the Index Column which is Created by reset_index()
written = written.drop('index',axis = 1)
#Printing the Merge DataFrame
print(written)
#Saving the DataFrame into Excel
written.to_excel("C:/Users/Akaash/Downloads/Q4_Updated.xlsx")
```

```
                cars  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, 4)
                  cars  Points  Score  Weigh
```

```
0   Cadillac Fleetwood    2.93   5.25  17.98
1  Lincoln Continental    3.00   5.42  17.82
```

```
                   cars  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    Cadillac Fleetwood    2.93  5.250  17.98
10  Lincoln Continental    3.00  5.420  17.82
```

5.Write a python program to scrape reviews from a commercial web site

[33]:

```
# import module
from bs4 import BeautifulSoup
import urllib.request
url = "https://www.geeksforgeeks.org/how-to-automate-an-excel-sheet-in-python/?
 ↪ref=feed"
#Use Request to open the url for Reading
openurl = urllib.request.urlopen(url)
#Creating Beautifulsoup Structure
soup = BeautifulSoup(openurl, 'html.parser')
# getting the Reviews
data = ''
for data in soup.find_all("div",{'class':['vote-d','vote-s']}):
    print(data.get_text())
```

```
Vote for difficulty
Current difficulty :
Basic
```

6.Create a 3x3 matrix with values ranging from 2 to 10 using numpy

[34]:

```
#Importing the Required library
import numpy as np
arr =  np.arange(2, 11).reshape(3,3)
print(arr)
```

```
[[ 2  3  4]
 [ 5  6  7]
 [ 8  9 10]]
```

7.Write a Python program to convert a list of numeric value into a one-dimensional NumPy array

[35]:

```
#Importing the Required library
import numpy as np
List = [12, 13, 100, 36]
print("Original List:",List)
arr = np.array(List)
print("One-dimensional NumPy array: ",arr)
print(type(arr))
```

```
Original List: [12, 13, 100, 36]
One-dimensional NumPy array:  [ 12  13 100  36]
<class 'numpy.ndarray'>
```

8.Write a Python program to create a null vector of size 10 and update sixth value to 11.

[36]:

```
#Importing the Required library
import numpy as np
arr = np.zeros(10)
print(arr)
print("Update sixth value to 11")
#Accessing & Updating 6th element
arr[6] = 11
print(arr)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Update sixth value to 11
[ 0.  0.  0.  0.  0.  0. 11.  0.  0.  0.]
```