Data Analysis and Visualization Practical

Name: Kuldeep Saini

College Roll No.: 20201415

Exam Roll No.: 20020570017

Semester: 5th

Subject: Data Analysis and Visualization

Course: BSc. (Hons) Computer Science

Q1

```
DL ={'Boys': [72, 68, 70, 69, 74], 'Girls': [63, 65, 69, 62, 61]}
```

```
import pandas as pd
pd.DataFrame(DL).to_dict(orient="records")
```

```
[{'Boys': 72, 'Girls': 63},
 {'Boys': 68, 'Girls': 65},
 {'Boys': 70, 'Girls': 69},
 {'Boys': 69, 'Girls': 62},
 {'Boys': 74, 'Girls': 61}]
```

Q2. a

```
import numpy as np  x = np.array([[10, 30],
[20, 60],[40,100]]) print("Mean of each
row:") print(x.mean(axis=1))
print("Standard
Deviation:") print(np.std(x,axis=1))
print("Variance:")
print(np.var(x,axis=1)
)
```

```
Mean of each row: [20. 40.
70.]
Standard Deviation: [10. 20.
30.] Variance:
[100. 400. 900.]
```

Q2. b

```
import numpy as np
    B = np.array( [56, 48, 22, 41, 78, 91, 24, 46, 8, 33])
    print("Original array:") print(B) i = np.argsort(B)
    print("Indices of the sorted elements of a given array:")
```

```
print(i)
```

```
Original array:
[56 48 22 41 78 91 24 46 8 33]
Indices of the sorted elements of a given array:
[8 2 6 9 3 7 1 0 4 5]
```

Q2.c

```
import numpy as np
R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries rowwise:")
for i in range(R): # A for loop for row entries a =[]
    for j in range(C):      # A for loop for column entries
        a.append(int(input()))
    matrix.append(a)
for i in range(R): for
j in range(C):  print(matrix[i][j],
        end = " ")
    print()
print(np.shape(matrix))
print(type(matrix)) newarray =
np.transpose(matrix) print(newarray)
```

```
Enter the number of rows:3
Enter the number of columns:4
Enter the entries rowwise:
1 2 3
4
5 6 6
5
4
3
2
1
1 2 3 4
5 6 6 5
4 3 2 1
(3, 4)
<class 'list'>
[[1 5 4]
 [2 6 3]  [3
 6 2]
 [4 5 1]]
```

```
import math as math arr =
[1,3,4,0,7,5,3,0,7,]   def find (arr):  return [i for i,
x in enumerate(arr) if x != 0 and not  math.isnan(x)]
def find_zero(arr): return [i for i , x in enumerate(arr) if x ==0]
```

```
arr1 = find(arr) arr2
=find_zero(arr)
print(arr1)
print(arr2)
```

```
[0, 1, 2, 4, 5, 6, 8]
[3, 7]
```

Q3

```
import pandas as pd import numpy as np  df =
pd.DataFrame(np.random.randint(0,100,size=(75,
4)), columns=list('ABCD')) df
```

```
        A B C D
0   85 67 26 95
1   12 23 23 17
2   11 18 18 72
3   92 4 33 80 4 78 55 20 56 .. .. .. .. ..
70  73 77 53 44
71  18 56 66 43
72  79 40 89 60
73  51 30 85 69
74  78 98 34 45

[75 rows x 4 columns] def
num_null(df):
    null_num = int(df.shape[0] * 0.1)  null_index =
    np.random.choice(df.index, null_num, replace=False)
    df.loc[null_index] = np.nan return df
num_null(df)
```

```
       A     B       C  D
0   85.0 67.0 26.0 95.0
1   12.0 23.0 23.0 17.0
2   11.0 18.0 18.0 72.0
3   92.0 4.0 33.0 80.0 4 78.0 55.0 20.0 56.0 .. ... ... ... ...
70   73.0 77.0 53.0 44.0
71   18.0 56.0 66.0 43.0
72   NaN NaN NaN NaN
73   51.0 30.0 85.0 69.0
74   78.0 98.0 34.0 45.0

[75 rows x 4 columns]
```

Q3. a

```
df.isnull().sum()
```

```
A    7
```

```
B    7
C    7
D    7
dtype: int64 df.isnull()
```

```
       A      B      C      D
0   False False False False
1   False False False False
2   False False False False
3   False False False False
4   False False False False
    ..  ...  ...  ...  ...
70   False False False False
71   False False False False
72   True True True True
73   False False False False
74   False False False False

[75 rows x 4 columns]
```

```
df['sum']=df.sum(axis=1) df.head()
```

```
      A     B        C  D  sum
0 85.0 67.0  26.0 95.0  273.0
1 12.0 23.0 23.0 17.0      75.0
2 11.0 18.0  18.0 72.0  119.0 3 92.0  4.0 33.0 80.0 209.0 4 78.0 55.0
  20.0 56.0  209.0
```

```
df.sort_values('sum',ascending=False)
```

```
       A     B        C  D  sum
18 77.0 97.0 64.0 47.0 285.0
39 75.0 46.0 72.0 92.0 285.0
44 79.0 93.0 85.0 24.0 281.0  0
85.0 67.0 26.0 95.0 273.0
33 94.0 54.0 48.0 69.0 265.0
25 66.0 97.0 9.0 92.0 264.0
22 32.0 56.0 97.0 79.0 264.0  5
20.0 85.0 72.0 78.0 255.0
26 83.0 34.0 74.0 62.0 253.0
49 44.0 43.0 63.0 99.0 249.0
30 96.0 33.0 66.0 53.0 248.0
70 73.0 77.0 53.0 44.0 247.0
61 24.0 41.0 93.0 86.0 244.0
7 67.0 52.0 26.0 99.0 244.0
50 36.0 22.0 94.0 89.0 241.0
23 69.0 20.0 82.0 67.0 238.0
73 51.0 30.0 85.0 69.0 235.0
12 27.0 73.0 62.0 70.0 232.0
```

```
52 59.0 99.0 39.0 32.0 229.0
27 7.0 90.0 46.0 76.0 219.0
6 53.0 79.0 44.0 37.0 213.0
34 64.0 32.0 18.0 96.0 210.0  3
92.0 4.0 33.0 80.0 209.0
4 78.0 55.0 20.0 56.0 209.0
14 91.0 4.0 97.0 14.0 206.0
21 48.0 70.0 12.0 75.0 205.0
9 52.0 40.0 86.0 26.0 204.0
19 3.0 87.0 35.0 75.0 200.0
38 82.0 35.0 82.0 1.0 200.0
54 98.0 35.0 33.0 27.0 193.0
24 60.0 28.0 27.0 76.0 191.0
71 18.0 56.0 66.0 43.0 183.0
36 15.0 63.0 48.0 53.0 179.0  8
93.0 7.0 54.0 25.0 179.0
67 45.0 71.0 13.0 48.0 177.0
32 72.0 30.0 15.0 55.0 172.0  45
4.0 62.0 21.0 85.0 172.0
48 35.0 31.0 82.0 22.0 170.0
41 40.0 44.0 48.0 38.0 170.0  10
1.0 19.0 64.0 86.0 170.0
68 21.0 51.0 63.0 34.0 169.0  16
58.0 58.0 24.0 20.0 160.0
17 71.0 4.0 6.0 74.0 155.0 20
6.0 14.0 94.0 40.0 154.0
53 49.0 31.0 23.0 38.0 141.0
69 26.0 55.0 18.0 40.0 139.0 11
87.0 11.0 38.0 1.0 137.0
64 10.0 17.0 76.0 31.0 134.0 28
29.0 79.0 5.0 7.0 120.0
2 11.0 18.0 18.0 72.0 119.0
55 65.0 24.0 9.0 20.0 118.0
29 12.0 93.0 7.0 2.0 114.0
13 56.0 20.0 31.0 7.0 114.0 60 0.0
27.0 67.0 5.0 99.0
    15 NaN NaN NaN NaN        0.0
```

```
df.drop(18,inplace=True) df
```

```
       A     B        C  D  sum
0 85.0 67.0 26.0 95.0  273.0
2 11.0 18.0 18.0 72.0  119.0 3
92.0 4.0 33.0 80.0 209.0 4 78.0 55.0
20.0 56.0  209.0
5   20.0 85.0 72.0 78.0  255.0
6   53.0 79.0 44.0 37.0  213.0
7   67.0 52.0 26.0 99.0  244.0 8 93.0 7.0 54.0 25.0 179.0 9 52.0 40.0
    86.0 26.0  204.0
10   1.0 19.0 64.0 86.0 170.0
```

```
11    87.0 11.0  38.0  1.0 137.0  12 27.0 73.0 62.0
      70.0 232.0
13    56.0 20.0  31.0  7.0 114.0
14    91.0  4.0 97.0 14.0 206.0
15    NaN NaN NaN NaN 0.0
16    58.0 58.0 24.0 20.0 160.0
17    71.0  4.0 6.0 74.0 155.0
19    3.0 87.0 35.0 75.0 200.0
20    6.0 14.0 94.0 40.0 154.0
21    48.0 70.0 12.0 75.0 205.0
22    32.0 56.0 97.0 79.0 264.0
23    69.0 20.0 82.0 67.0 238.0
24    60.0 28.0 27.0 76.0 191.0
25    66.0 97.0  9.0 92.0  264.0
26    83.0 34.0 74.0 62.0 253.0
27    7.0 90.0 46.0 76.0 219.0
28    29.0 79.0  5.0 7.0 120.0
29    12.0 93.0  7.0 2.0 114.0
30    96.0 33.0 66.0 53.0 248.0
32 72.0 30.0 15.0 55.0 172.0  33
94.0 54.0 48.0 69.0 265.0
34  64.0  32.0  18.0  96.0  210.0
36  15.0  63.0  48.0  53.0  179.0
38  82.0  35.0  82.0   1.0  200.0
39  75.0  46.0  72.0  92.0  285.0
41  40.0  44.0  48.0  38.0  170.0
44  79.0  93.0  85.0  24.0  281.0
45   4.0  62.0  21.0  85.0  172.0
48  35.0  31.0  82.0  22.0  170.0
49  44.0  43.0  63.0  99.0  249.0
50  36.0  22.0  94.0  89.0  241.0
52  59.0  99.0  39.0  32.0  229.0
53  49.0  31.0  23.0  38.0  141.0
54  98.0  35.0  33.0  27.0  193.0
55  65.0  24.0   9.0  20.0  118.0
60   0.0  27.0  67.0   5.0   99.0
61  24.0  41.0  93.0  86.0  244.0
64  10.0  17.0  76.0  31.0  134.0
67  45.0  71.0  13.0  48.0  177.0
68  21.0  51.0  63.0  34.0  169.0
69  26.0  55.0  18.0  40.0  139.0
70  73.0  77.0  53.0  44.0  247.0
71  18.0  56.0  66.0  43.0  183.0
73  51.0  30.0  85.0  69.0  235.0
```

Q3.c

```
mod_df = df.dropna( axis=0,
                thresh=5)
```

```
mod_df
```

```
        A     B        C  D  sum
0 85.0 67.0 26.0 95.0 273.0
2   11.0 18.0 18.0 72.0 119.0
3   92.0 4.0 33.0 80.0 209.0
4   78.0 55.0 20.0 56.0 209.0
5   20.0 85.0 72.0 78.0 255.0 .. ... ... ... ... ...
69  26.0 55.0 18.0 40.0 139.0
70  73.0 77.0 53.0 44.0 247.0
71  18.0 56.0 66.0 43.0 183.0
73  51.0 30.0 85.0 69.0 235.0
74  78.0 98.0 34.0 45.0 255.0

[67 rows x 5 columns]
```

Q3. d

```
sort_col = df.sort_values(by= 'A',ascending=False) sort_col
```

```
        A     B        C  D  sum
54 98.0 35.0 33.0 27.0 193.0
30 96.0 33.0 66.0 53.0 248.0
33 94.0 54.0 48.0 69.0 265.0
8 93.0 7.0 54.0 25.0 179.0
3 92.0 4.0 33.0 80.0 209.0
14 91.0 4.0 97.0 14.0 206.0
11 87.0 11.0 38.0 1.0 137.0
0 85.0 67.0 26.0 95.0 273.0
26 83.0 34.0 74.0 62.0 253.0
38 82.0 35.0 82.0 1.0 200.0
44 79.0 93.0 85.0 24.0 281.0  4
78.0 55.0 20.0 56.0 209.0
18 77.0 97.0 64.0 47.0 285.0
39 75.0 46.0 72.0 92.0 285.0
70 73.0 77.0 53.0 44.0 247.0
32 72.0 30.0 15.0 55.0 172.0  17
71.0 4.0 6.0 74.0 155.0
23 69.0 20.0 82.0 67.0 238.0
7 67.0 52.0 26.0 99.0 244.0
25 66.0 97.0 9.0 92.0 264.0
55 65.0 24.0 9.0 20.0 118.0
34 64.0 32.0 18.0 96.0 210.0
24 60.0 28.0 27.0 76.0 191.0
52 59.0 99.0 39.0 32.0 229.0
16 58.0 58.0 24.0 20.0 160.0
13 56.0 20.0 31.0 7.0 114.0
6 53.0 79.0 44.0 37.0 213.0
9 52.0 40.0 86.0 26.0 204.0
73 51.0 30.0 85.0 69.0 235.0
```

```
53 49.0 31.0 23.0 38.0 141.0
21 48.0 70.0 12.0 75.0 205.0
67 45.0 71.0 13.0 48.0 177.0
49 44.0 43.0 63.0 99.0 249.0
41 40.0 44.0 48.0 38.0 170.0
50 36.0 22.0 94.0 89.0 241.0
48 35.0 31.0 82.0 22.0 170.0
22 32.0 56.0 97.0 79.0 264.0  28
29.0 79.0 5.0 7.0 120.0
12 27.0 73.0 62.0 70.0 232.0
69 26.0 55.0 18.0 40.0 139.0
61 24.0 41.0 93.0 86.0 244.0
68 21.0 51.0 63.0 34.0 169.0  5
20.0 85.0 72.0 78.0 255.0
71 18.0 56.0 66.0 43.0 183.0
36 15.0 63.0 48.0 53.0 179.0
29 12.0 93.0 7.0 2.0 114.0
2 11.0 18.0 18.0 72.0 119.0
64 10.0 17.0 76.0 31.0 134.0
27 7.0 90.0 46.0 76.0 219.0
20 6.0 14.0 94.0 40.0 154.0
45 4.0 62.0 21.0 85.0 172.0 19 3.0 87.0 35.0 75.0 200.0


10 1.0 19.0 64.0 86.0 170.0 60 0.0
27.0 67.0 5.0 99.0
    15 NaN NaN NaN NaN        0.0
```

Q3. e

```
df.drop_duplicates(subset='A',keep='first',inplace=True) df
```

```
       A     B        C  D  sum
0 85.0 67.0 26.0 95.0 273.0
2    11.0 18.0 18.0 72.0 119.0
3    92.0 4.0 33.0 80.0 209.0
4    78.0 55.0 20.0 56.0 209.0
5    20.0 85.0 72.0 78.0 255.0
6    53.0 79.0 44.0 37.0 213.0
7    67.0 52.0 26.0 99.0 244.0
8    93.0 7.0 54.0 25.0 179.0
9    52.0 40.0 86.0 26.0 204.0
10   1.0 19.0 64.0 86.0 170.0
11   87.0 11.0 38.0 1.0 137.0
12   27.0 73.0 62.0 70.0 232.0
13   56.0 20.0 31.0 7.0 114.0
14   91.0 4.0 97.0 14.0 206.0 15 NaN NaN NaN NaN 0.0
16   58.0 58.0 24.0 20.0 160.0
```

17 71.0 4.0 6.0 74.0 155.0
18 77.0 97.0 64.0 47.0 285.0
19 3.0 87.0 35.0 75.0 200.0
20 6.0 14.0 94.0 40.0 154.0
21 48.0 70.0 12.0 75.0 205.0
22 32.0 56.0 97.0 79.0 264.0
23 69.0 20.0 82.0 67.0 238.0
24 60.0 28.0 27.0 76.0 191.0
25 66.0 97.0 9.0 92.0 264.0
26 83.0 34.0 74.0 62.0 253.0
27 7.0 90.0 46.0 76.0 219.0
28 29.0 79.0 5.0 7.0 120.0
29 12.0 93.0 7.0 2.0 114.0
30 96.0 33.0 66.0 53.0 248.0
32 72.0 30.0 15.0 55.0 172.0
33 94.0 54.0 48.0 69.0 265.0
34 64.0 32.0 18.0 96.0 210.0
36 15.0 63.0 48.0 53.0 179.0 38
82.0 35.0 82.0 1.0 200.0
39 75.0 46.0 72.0 92.0 285.0
41 40.0 44.0 48.0 38.0 170.0
44 79.0 93.0 85.0 24.0 281.0
45 4.0 62.0 21.0 85.0 172.0
48 35.0 31.0 82.0 22.0 170.0

49	44.0	43.0	63.0	99.0	249.0
50	36.0	22.0	94.0	89.0	241.0
52	59.0	99.0	39.0	32.0	229.0
53	49.0	31.0	23.0	38.0	141.0
54	98.0	35.0	33.0	27.0	193.0
55	65.0	24.0	9.0	20.0	118.0
60	0.0	27.0	67.0	5.0	99.0
61	24.0	41.0	93.0	86.0	244.0
64	10.0	17.0	76.0	31.0	134.0
67	45.0	71.0	13.0	48.0	177.0
68	21.0	51.0	63.0	34.0	169.0
69	26.0	55.0	18.0	40.0	139.0
70	73.0	77.0	53.0	44.0	247.0
71	18.0	56.0	66.0	43.0	183.0
73	51.0	30.0	85.0	69.0	235.0

Q3. f

```
column_1 = df["A"] column_2 = df["B"]
correlation = column_1.corr(column_2)
correlation
-0.1914983464702535 print(df.B.cov(df.C))
```

-161.4713487071978

Q4

```
xls1=pd.ExcelFile('/content/Attendance1.xlsx'
) xls1.sheet_names f1=pd.read_excel(xls1,xls1.sheet_names[0])
f1 xls2=pd.ExcelFile('/content/Attendance2.xlsx'
)                              xls2.sheet_names
f2=pd.read_excel(xls2,xls2.sheet_names[0]) f2
```

Q4. a

```
j=pd.merge(f1,f2,on=['Name'])  j['Name']
```

Q4. b

```
k=pd.merge(f1,f2,how='outer',on=['Name'])
k['Name']
```

Q4. c

```
frames=[f1,f2]     result=pd.concat(frames,
keys=['f1', 'f2']) result
```

Q4. d

```
f_new=pd.merge(f1,f2)
df2=f_new.set_index(keys=[f_new.columns[0],f_new.columns[2]]
) df2 df2.describe()
```

Q5

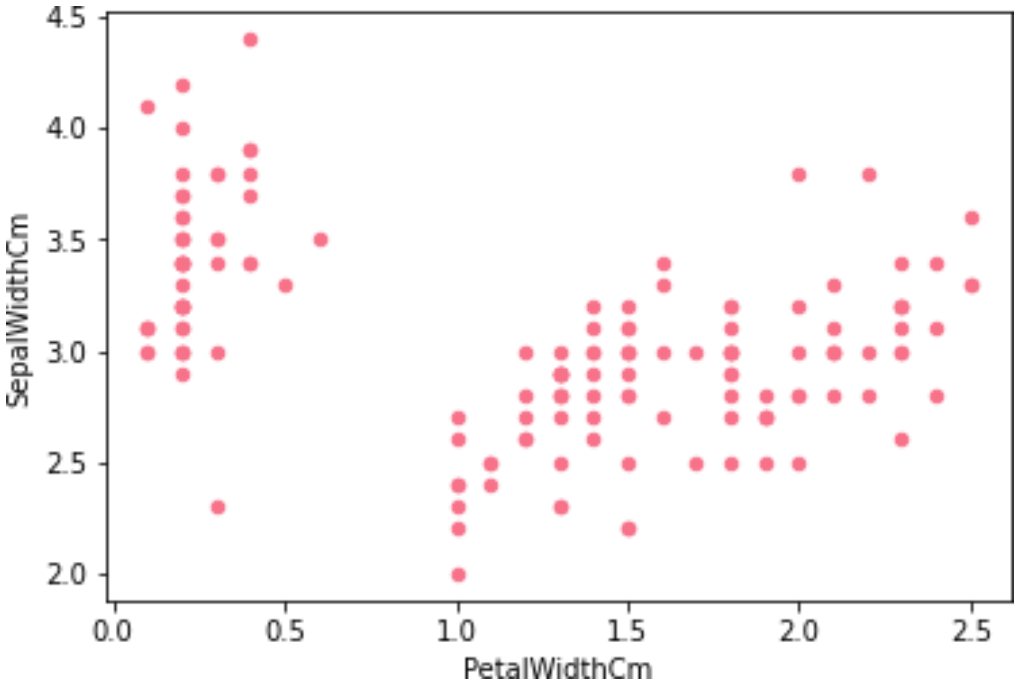
```
import numpy as np import
pandas as pd import seaborn as
sns sns.set_palette('husl')
import matplotlib.pyplot as plt
%matplotlib inline data =
pd.read_csv('Iris.csv') data.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris- setosa
1	2	4.9	3.0	1.4	0.2	Iris- setosa
2	3	4.7	3.2	1.3	0.2	Iris- setosa
3	4	4.6	3.1	1.5	0.2	Iris- setosa

```
4     5 5.0      3.6  1.4  0.2  Iris- setosa
data.plot(kind='scatter', x='PetalWidthCm',y='SepalWidthCm') plt.show
```

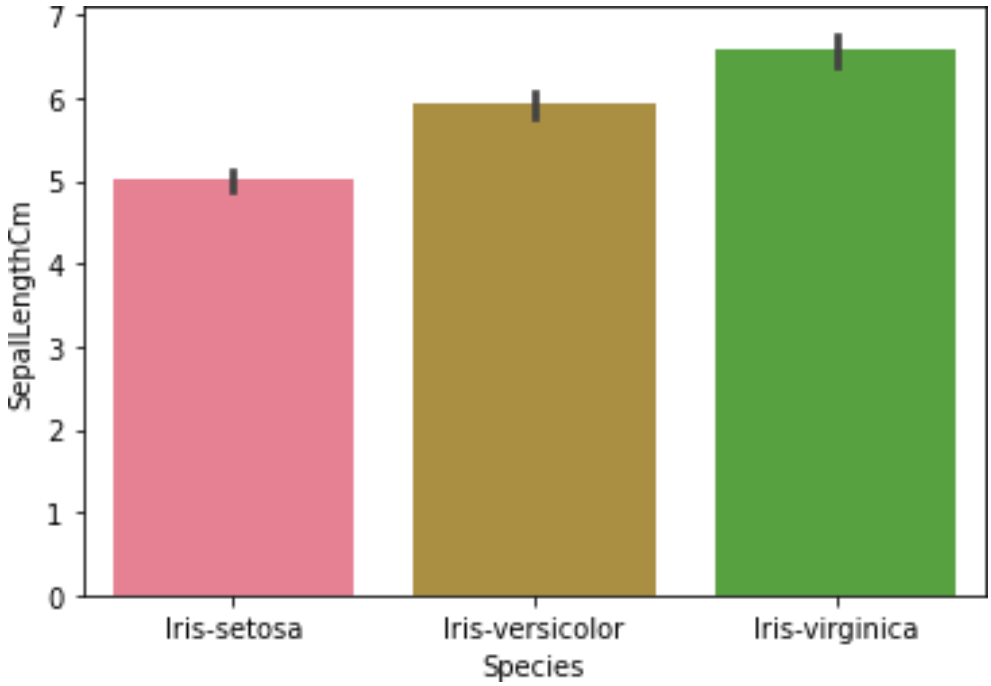
```
*c* argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with *x* & *y*. Please use the *color* keyword-
argument or provide a 2-D array with a single row if you intend to
specify the same RGB or RGBA value for all points.
```

```
<function matplotlib.pyplot.show>
```



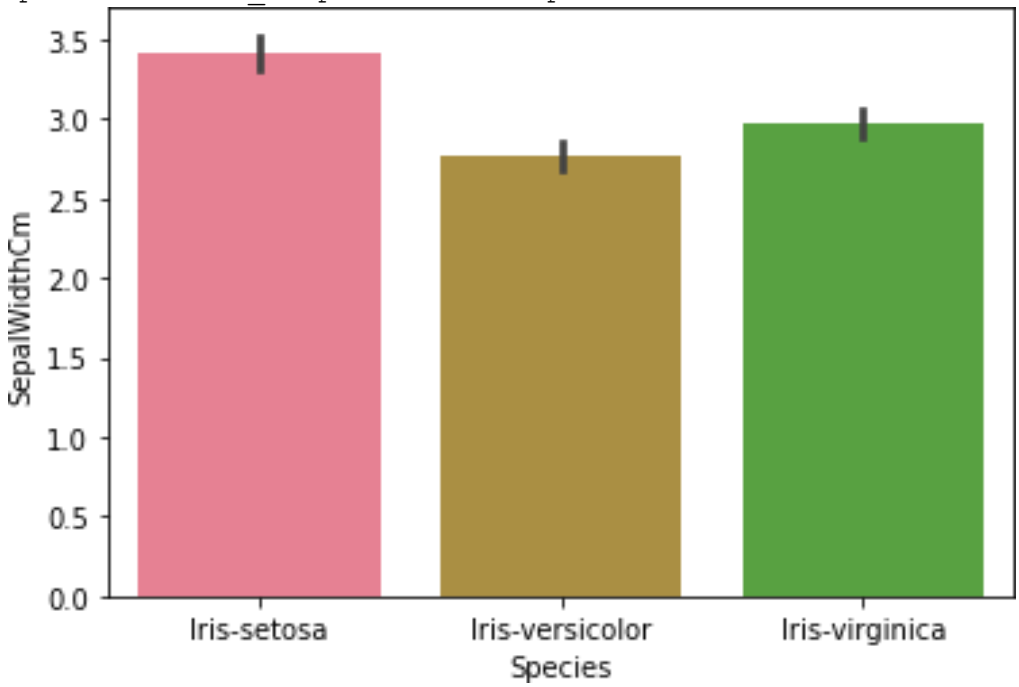

```
sns.barplot(x='Species',y='SepalLengthCm',data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff64c763310>
```



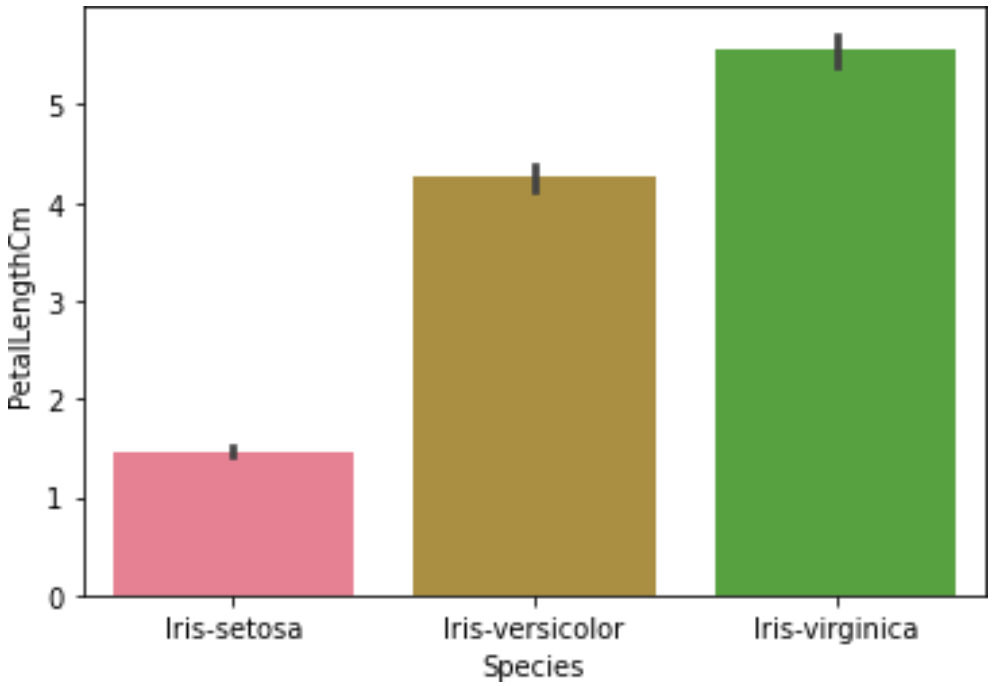

```
sns.barplot(x='Species',y='SepalWidthCm',data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff64c263450>
```



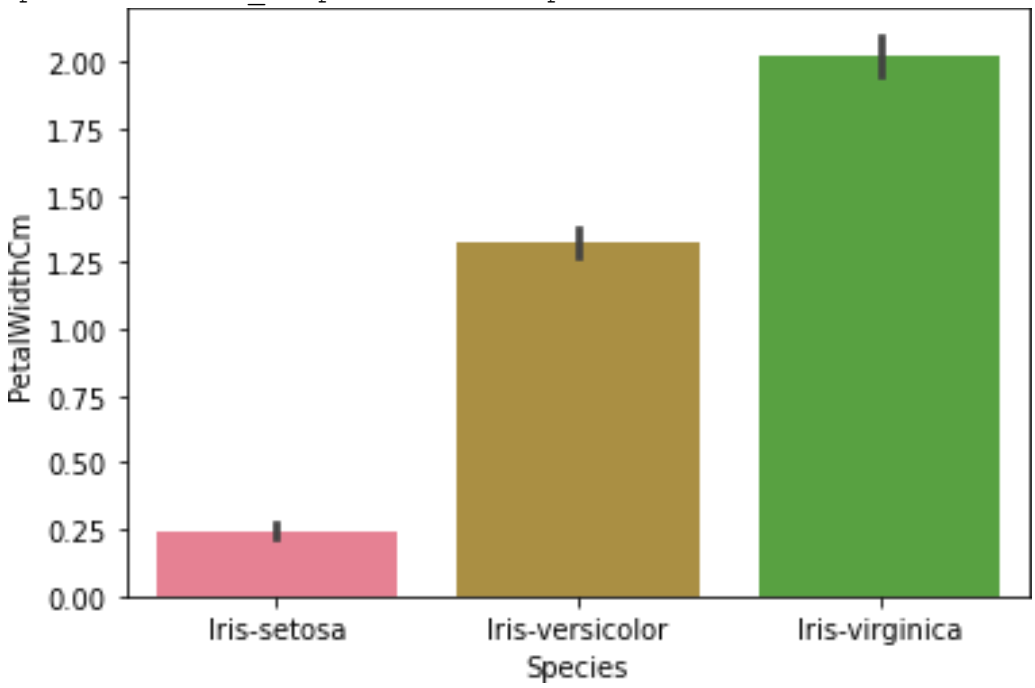

```
sns.barplot(x='Species',y='PetalLengthCm',data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff64c1ddc10>
```

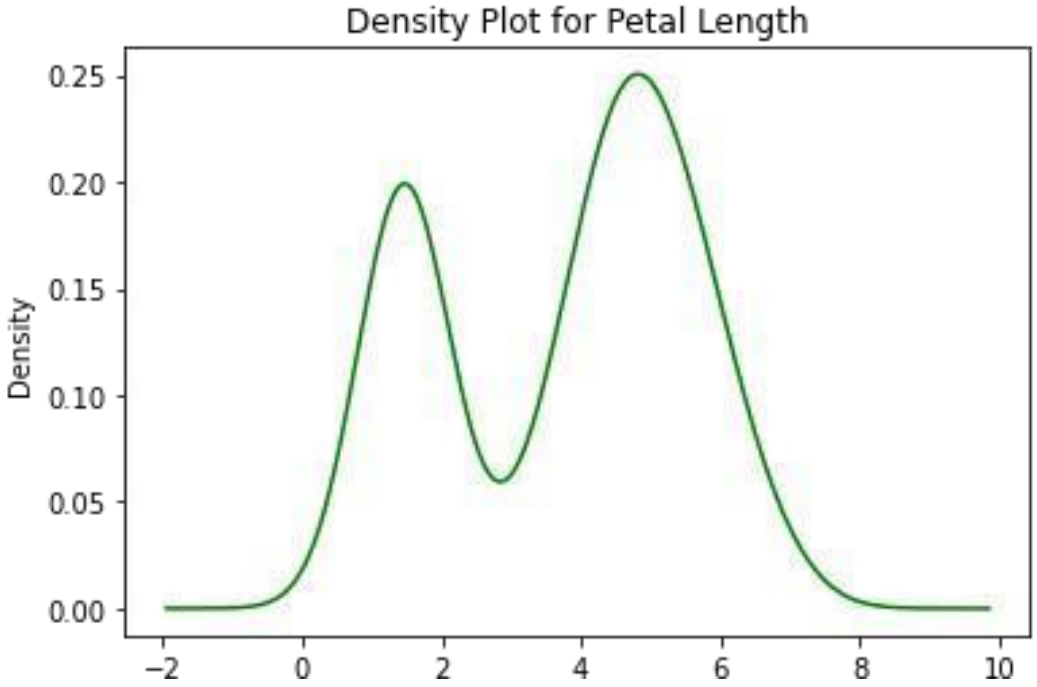



```
sns.barplot(x='Species',y='PetalWidthCm',data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff64c14df50>
```

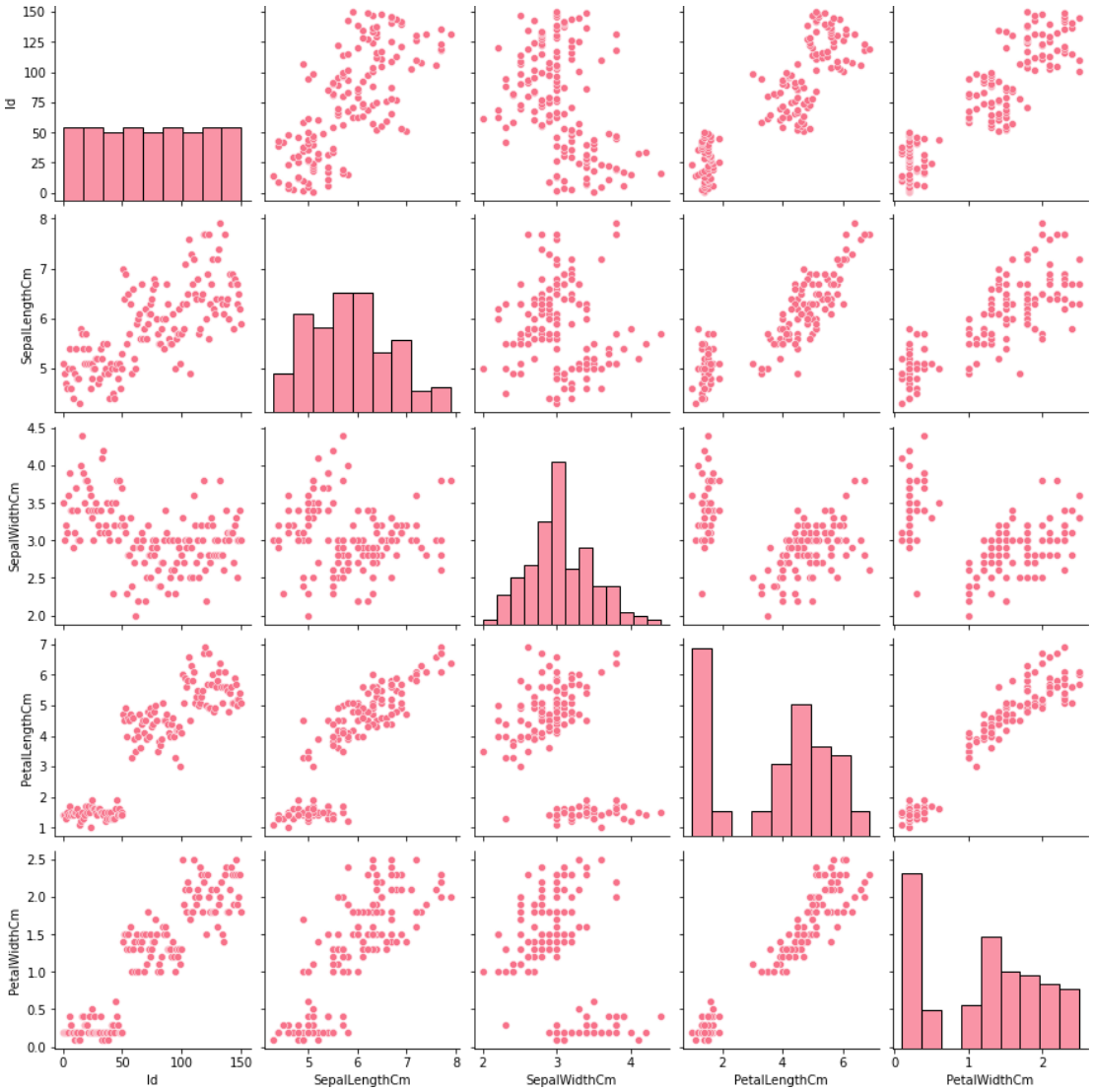



```
 data.PetalLengthCm.plot.density(color='green')
plt.title('Density Plot for Petal Length')
plt.show()
```




```
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x7ff64c6c3990>
```



Q6

```
data2=
pd.read_csv('https://raw.githubusercontent.com/codebasics/py/master/
pandas/14_ts_datetimeindex/aapl.csv') data2.head(10)
```

```
          Date  Open  High          Low Close  Volume
0   7-Jul-17 142.90 144.75 142.90 144.18 19201712
1   6-Jul-17 143.02 143.50 142.41 142.73 24128782
2   5-Jul-17 143.69 144.79 142.72 144.09 21569557
3   3-Jul-17 144.88 145.30 143.10 143.50 14277848
4   30-Jun-17 144.45 144.96 143.78 144.02 23024107
5   29-Jun-17 144.71 145.13 142.28 143.68 31499368
6   28-Jun-17 144.49 146.11 143.16 145.83 22082432
7   27-Jun-17 145.01 146.16 143.62 143.73 24761891
8   26-Jun-17 147.17 148.28 145.38 145.82 25692361 9 23-Jun-17 145.13
    147.16 145.11 146.28 35439389
    data2.groupby('Open')['Volume'].mean()
Open
```

```
 96.75     23794945.0
 96.82     56239822.0
97.17      24167463.0
97.39      38918997.0
 97.41     25892171.0
              ...
155.02     21069647.0
155.19     64882657.0
155.25     21250798.0
155.94     20048478.0
156.01     26009719.0
Name: Volume, Length: 246, dtype: float64 data2.groupby('Open',

as_index=False)['Volume'].mean()


       Open      Volume
0     96.75 23794945.0
1     96.82 56239822.0
2     97.17 24167463.0
3     97.39 38918997.0
4     97.41 25892171.0
..      ...         ...
241  155.02 21069647.0
242  155.19 64882657.0
243  155.25 21250798.0
244  155.94 20048478.0
245  156.01 26009719.0

[246 rows x 2 columns] data2['Date'] = pd.to_datetime(data2['Date'])

data2.head(10)

Date            Open  High  Low        Close Volume
0 2017-07-07 142.90 144.75 142.90 144.18 19201712
1 2017-07-06 143.02 143.50 142.41 142.73 24128782
2 2017-07-05 143.69 144.79 142.72 144.09 21569557
3 2017-07-03 144.88 145.30 143.10 143.50 14277848
4 2017-06-30 144.45 144.96 143.78 144.02 23024107
5 2017-06-29 144.71 145.13 142.28 143.68 31499368
6 2017-06-28 144.49 146.11 143.16 145.83 22082432
7 2017-06-27 145.01 146.16 143.62 143.73 24761891
8 2017-06-26 147.17 148.28 145.38 145.82 25692361
9 2017-06-23 145.13 147.16 145.11 146.28 35439389 df_agg =
  data2.groupby(['High','Low']).agg({'Volume':sum}) result =
  df_agg['Volume'].groupby(level=0, group_keys=False)
  print(result.nlargest())
```

```
High  Low
97.65 96.73       23794945
97.67 96.84       25892171
97.70 97.12       24167463
97.97 96.42       56239822
98.84 96.92       40382921
                    ...
155.81 153.78     26624926
155.98 154.48     21069647
156.06 154.72     20048478
156.42 154.67     32527017
156.65 155.05     26009719
Name: Volume, Length: 251, dtype: int64
```

```
groups = data2.groupby(['Close', pd.cut(data2.Open,
3)]) result = groups.size().unstack() print(result)
```

```
Open     (96.691, 116.503] (116.503, 136.257] (136.257, 156.01]
Close

96.67                    1                    0                   0

96.87                    1                    0                   0

96.98                    1                    0                   0

97.34                    1                    0                   0

97.42  1  0  0   ...  ...  ...  ...

155.37                   0                    0                   1

155.45                   0                    0                   1

155.47                   0                    0                   1

155.70                   0                    0                   1

156.10                   0                    0                   1


[239 rows x 3 columns]
```

Q7

```
 df3 =
pd.DataFrame({
'Name':['Mudit Chauhan','Seema Chopra','rani gupta','aditya
narayan','sanjeev sahani','prakash kumar','Ritu Agarwal','Akshay
Goel','Meeta Kulkarni','Preeti Ahuja','Sunil Das Gupta','Sonali
Sapre','Rashmi Talwar','Ashish Dubey','Kiran Sharma','Sameer
Bansal'],
'Birth_Month':
['December','January','March','October','February','December','Septemb
er','August','July','November','April','January','May','June','Februa
r y','October'],
```

```
'Gender':
['M','F','F','M','M','M','F','M','F','F','M','F','F','M','F','M'],
'Pass_division':[3,2,1,1,2,3,1,1,2,2,3,1,3,2,2,1]})
 df3
Nam
e
Bir
th_
Mon
th
Gen
der
Pas
s_d
ivi
sio
n
0        Mudit Chauhan     December  M  3
1        Seema Chopra      January  F  2
2        rani gupta  March       F  1
3        aditya narayan  October  M  1
4        sanjeev sahani  February  M  2
5        prakash kumar     December  M  3
6        Ritu Agarwal September  F  1
7        Akshay Goel  August  M  1
8        Meeta Kulkarni  July  F  2
9        Preeti Ahuja      November  F  2
10       Sunil Das Gupta  April       M  3
11       Sonali Sapre      January  F  1
12       Rashmi Talwar     May  F  3
13       Ashish Dubey      June  M  2
14       Kiran Sharma      February  F  2
15       Sameer Bansal     October  M  1
pd.get_dummies(df3.Gender)
F M  0 0 1
1   1 0
2   1 0
3   0 1
4   0 1
5   0 1
```

```
      6   1 0
      7   0 1
      8   1 0
      9   1 0
      10  0 1
      11  1 0
      12  1 0
      13  0 1  14 1 0 15 0 1
```

```
pd.get_dummies(df3.Gender, drop_first=True)
```

```
    M  0
1
1   0
2   0
3   1
4   1
5   1
6   0
7   1
8   0
9   0
10  1
11  0
12  0
13  1
14  0 15 1
```

```
gender_dummies = pd.get_dummies(df3.Gender, prefix='Gender')
gender_dummies
```

```
    Gender_F Gender_M
0          0    1
1          1  0
2          1  0
3          0  1
4          0  1
5          0  1
6          1  0
```

```
7          0  1
8          1  0
9          1  0
10         0  1
11         1  0
12         1  0
13         0  1
14         1  0
15         0  1
```

```
df3 = pd.concat([df3, gender_dummies], axis=1) df3.head()
```

```
             Name Birth_Month Gender Pass_division Gender_F
Gender_M
0       Mudit Chauhan  December  M  3  0
1
1       Seema Chopra  January  F  2  1
0
2       rani gupta  March  F  1  1
0
3       aditya narayan     October  M  1  0
1
4       sanjeev sahani     February  M  2  0
1
```

```
pass_dummies = pd.get_dummies(df3.Pass_division, prefix='pass')
pass_dummies.head() pass_1
   pass_2 pass_3 0
0       0 1
1       0  1  0
2       1  0  0
3       1  0  0
4       0  1  0
```

```
df3 = pd.concat([df3, pass_dummies], axis=1)
df3.head()
```

```
             Name Birth_Month Gender ... pass_1 pass_2 pass_3
0      Mudit Chauhan  December  M ... 0  0  1
1      Seema Chopra  January  F ... 0  1  0
2      rani gupta  March  F ... 1  0  0
```

```
3      aditya narayan  October  M ... 1  0  0 4 sanjeev sahani
       February  M ... 0  1  0

5       rows x 9 columns]
                    df3

                    Name Birth_Month Gender ... pass_1 pass_2 pass_3
 0   Mudit Chauhan   December  M ... 0  0  1
 1   Seema Chopra    January  F ... 0  1  0
 2   rani gupta  March     F ... 1  0  0
 3   aditya narayan  October  M ... 1  0  0
 4   sanjeev sahani  February  M ... 0  1  0
 5   prakash kumar   December  M ... 0  0  1
 6   Ritu Agarwal September  F ... 1  0  0
 7   Akshay Goel  August  M ... 1  0  0
 8   Meeta Kulkarni  July  F ... 0  1  0
 9   Preeti Ahuja    November  F ... 0  1  0
 10  Sunil Das Gupta  April     M ... 0  0  1
 11  Sonali Sapre    January  F ... 1  0  0
 12  Rashmi Talwar   May  F ... 0  0  1
 13  Ashish Dubey    June  M ... 0  1  0
 14  Kiran Sharma    February  F ... 0  1  0
 15  Sameer Bansal   October  M ... 1  0  0
[16 rows x 9 columns] df3.sort_values(by='Birth_Month')

  Name                Birth_Month Gender ... pass_1 pass_2 pass_3
10 Sunil Das Gupta April        M ...        0      0       1
 7 Akshay Goel      August       M ...        1      0       0
  0 Mudit Chauhan December      M ...        0      0       1
  5 prakash kumar December      M ...        0      0       1
 4 sanjeev sahani February      M ...        0      1       0
   14 Kiran Sharma  February  F ...          0      1       0
 1 Seema Chopra     January      F ...        0      1       0

11       Sonali Sapre  January       F ...        1       0       0
 8  Meeta Kulkarni          July      F ...        0       1       0
13     Ashish Dubey         June      M ...        0       1       0
2        rani gupta       March      F ...        1       0       0
 12  Rashmi Talwar           May      F ...        0       0       1
9        Preeti Ahuja  November       F ...        0       1       0
```

```
 3  aditya narayan      October      M ...         1       0       0
 15  Sameer Bansal      October      M ...         1       0       0
6       Ritu Agarwal September   F ...      1     0     0


[16 rows x 9 columns]
```

```
sort_order =
['January','February','March','April','May','June','July','August','S
e ptember','October','November','December'] df3.index =
pd.CategoricalIndex(df3['Birth_Month'],
categories=sort_order,ordered=True) df3 =
df3.sort_index().reset_index(drop=True) df3
```

```
                    Name Birth_Month Gender ... pass_1 pass_2 pass_3
 0    Seema Chopra     January  F ... 0  1  0
 1    Sonali Sapre     January  F ... 1  0  0
 2    sanjeev sahani  February  M ... 0  1  0
 3    Kiran Sharma     February  F ... 0  1  0
 4    rani gupta  March     F ... 1  0  0
 5    Sunil Das Gupta  April     M ... 0  0  1
 6    Rashmi Talwar    May  F ... 0  0  1
 7    Ashish Dubey     June  M ... 0  1  0
 8    Meeta Kulkarni  July  F ... 0  1  0
 9    Akshay Goel  August  M ... 1  0  0
 10  Ritu Agarwal September  F ... 1  0  0
 11  aditya narayan  October  M ... 1  0  0
 12  Sameer Bansal    October  M ... 1  0  0
 13  Preeti Ahuja     November  F ... 0  1  0
 14  Mudit Chauhan    December  M ... 0  0  1
 15  prakash kumar    December  M ... 0  0  1
[16 rows x 9 columns]
```

Q8

```
df4= pd.DataFrame({
'Name':
['Shah','Vats','Vats','Kumar','Vats','Kumar','Shah','Shah','Kumar','Sh
ah'],
'Gender':
['Male','Male' ,'Female','Female','Female','Male','Male','Female','Fem
ale','Male'],
'Monthly_Income (Rs)':
[114000,65000,43150,69500,155000,103000,55000,112400,81030,71900]})
 df4
```

```
    Name Gender Monthly_Income (Rs)
0   Shah  Male  114000
```

```
1   Vats  Male  65000
2   Vats Female  43150
3   Kumar Female      69500
4   Vats Female  155000
5   Kumar  Male  103000
6   Shah  Male  55000
7   Shah Female  112400
8   Kumar Female      81030
9   Shah  Male  71900
```

```
sumOfIncome = df4.groupby(by=['Name'],
as_index=False)['Monthly_Income (Rs)'].sum()
print (sumOfIncome)
```

```
    Name Monthly_Income (Rs)
0   Kumar 253530
1   Shah 353300  2 Vats 263150
```

```
grouped = df4.groupby(['Name'], sort=False)['Monthly_Income
(Rs)'].max() print(grouped)
```

```
Name
Shah     114000
Vats     155000
Kumar      103000  Name: Monthly_Income (Rs),
dtype: int64  res = df4[df4['Monthly_Income
(Rs)'] > 60000] res
```

```
    Name Gender Monthly_Income (Rs)
1   Shah  Male 114000
2   Vats  Male 65000
3   Kumar Female      69500
4   Vats Female 155000
5   Kumar  Male 103000
7   Shah Female 112400
8   Kumar Female      81030 9 Shah  Male 71900
```

```
res4 = df4[(df4['Name'] == 'Shah') &
          (df4['Gender'] == 'Female')]  res4.mean()
Monthly_Income (Rs) 112400.0 dtype:
float64
```