Name : Abhishek Kumar

Course : Bsc (H) computer science

Section : A

Roll No. : 10814

Subject : Data Analysis And Visualization

PRACTICAL LIST SOLUTION:

Question1

CODE:

```
original_dict = {'boys':[72, 68, 70, 69, 74], 'girls':[63, 65, 69, 62, 61]}

empty_list = []

for i in range(len(original_dict['boys'])):

 empty_list.append({'boys':original_dict['boys'][i],'girls':original_dict['girls'][i]})

print(empty_list)
```

OUTPUT:

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

Question2

CODE:

```
import numpy as np

arr1 = np.random.randint(0, 200, size=(3,4))    #generation of randon array

print(arr1)
```

OUTPUT:

```
[[126  27  79 186]
 [  0 169 169 156]
 [ 21 176 193 182]]
```

CODE:

```
#Part - a

#mean

mean_arr = np.mean(arr1, axis=1)

print("Mean : ",mean_arr)

#standard Deviation

std_arr= np.std(arr1, axis=1)

print("Standard Daviation : ",std_arr)

#Variance

variance_arr = np.var(arr1, axis=1)

print("variance : ",variance_arr)
```

OUTPUT:

```
Mean :  [104.5 123.5 143. ]
Standard Daviation :  [58.65364439 71.5        70.70007072]
variance :  [3440.25 5112.25 4998.5 ]
```

CODE:

```
#Part - b

B = np.array([56, 48, 22, 41, 78, 91, 24, 46, 8, 33])

indices = np.argsort(B)

print( "indexes of elements of array in a sorted manner : ",indices)
```

OUTPUT:

```
indexes of elements of array in a sorted manner :  [8 2 6 9 3 7 1 0 4 5]
```

CODE:

```
#Part - c

m = int(input("Enter the number of rows(m) : "))

n = int(input("Enter the number of columns(n) : "))

arr2 = np.random.randint(0, 100, size=(m, n))
```

```
print(arr2)

print("Shape of array  :", arr2.shape)

print("Type of array  :", type(arr2))

print("Data type :", arr2.dtype)

reshaped_array = arr2.reshape(n, m)

print("Reshaped array :", reshaped_array)
```

OUTPUT:

```
Enter the number of rows(m) : 2
Enter the number of columns(n) : 2
[[ 1 40]
 [52 52]]
Shape of array  : (2, 2)
Type of array  : <class 'numpy.ndarray'>
Data type : int64
Reshaped array : [[ 1 40]
 [52 52]]
```

CODE:

```
#Part - d

arr=np.array([0,9,0,38,4,"NaN",22,0,36,88,82,29,64,"Nan"])

zero1=[]

nan1=[]

nonzero1=[]


print("Original array : ",arr)

for i in range(len(arr)):

    if(arr[i].isalpha()):

        nan1.append(i)

    elif(arr[i]=='0'):

        zero1.append(i)

    elif(arr[i]!='0'):

        nonzero1.append(i)


print("Indices of NaN elements are ", nan1)

print("Indices of zero elements are ", zero1)
```

```
print("Indices of non zero elements are", nonzero1)
```

OUTPUT:

```
Original array :  ['0' '9' '0' '38' '4' 'NaN' '22' '0' '36' '88' '82' '29' '64' 'Nan']
Indices of NaN elements are  [5, 13]
Indices of zero elements are  [0, 2, 7]
Indices of non zero elements are [1, 3, 4, 6, 8, 9, 10, 11, 12]
```

Question3

CODE:

```
import pandas as pd

import numpy as np

data=np.random.randint(1,100,size=(50,3))

df=pd.DataFrame(data,columns=["First column","Second column","Third column"])

df
```

OUTPUT:

	First column	Second column	Third column
0	19	14	79
1	26	49	86
2	28	95	45
3	50	28	61
4	87	54	29
5	72	94	98
6	97	46	6
7	64	29	26
8	60	64	23
9	43	13	50
10	65	88	39
11	49	81	53
12	51	70	64

18	38	12	48
19	44	47	67
20	45	8	73
21	36	55	87
22	53	47	49
23	44	21	49
24	77	52	50
25	90	85	57
26	36	16	37
27	42	86	45
28	59	99	36
29	78	31	66
30	95	10	38
31	52	12	94

CODE:

```
r1= 50

c1= 3

data=np.random.randn(r1,c1)

null_index= np.random.choice([True,False],size=(r1,c1),p=[0.10,0.90])

data[null_index]=np.nan

df=pd.DataFrame(data,columns=["First column","Second column","Third column"])

print(abs(df))
```

OUTPUT:

	First column	Second column	Third column
0	1.350895	0.955364	0.976909
1	2.313891	0.219313	0.549008
2	1.561012	0.209205	0.003799
3	NaN	0.287668	NaN
4	0.659472	1.740067	0.414668
5	0.193832	0.279870	NaN
6	0.567887	0.286749	0.949145
7	0.686564	0.490299	0.470725
8	0.260752	0.605505	0.525147
9	0.580914	2.882207	0.134727
10	0.214105	0.232941	NaN
11	0.709970	0.089840	0.542781
12	1.110686	1.483283	NaN
13	0.642417	0.750721	1.262332
14	0.859094	0.581935	1.444663
15	1.387090	0.584787	0.568641
16	1.432541	NaN	2.383367
17	1.574465	0.318741	0.471941
18	1.083654	0.246662	0.560072
19	1.710082	NaN	0.476621
20	1.249635	0.134312	1.668833
21	0.255575	0.219323	1.480682
22	1.111131	0.068021	0.414351
23	NaN	0.757836	0.317410
24	0.845499	0.921209	1.580022
28	0.264037	NaN	1.067192
29	NaN	0.781102	0.195623
30	0.428327	0.249611	1.208994
31	1.527816	0.977799	1.046835
32	0.854621	0.324819	0.859144
33	0.180664	0.948092	2.783761
34	2.304631	0.981199	0.561495
35	0.015376	1.031072	1.798057
36	NaN	NaN	0.070108
37	0.034891	0.791883	0.577948
38	1.505285	0.686674	0.944063
39	0.663310	0.431315	0.514068
40	0.741185	0.718974	0.549174
41	0.382114	0.590958	0.332176
42	0.731859	0.474825	0.485785
43	NaN	0.228828	1.953285
44	0.084395	0.986810	1.936875
45	0.386071	1.124862	0.659375
46	0.554185	0.504691	1.285035
47	0.073791	2.770845	0.759759
48	2.424553	NaN	1.230991
49	0.755168	2.111597	1.116611

CODE:

```
#Part a

df1=df.isnull().sum()

df1
```

OUTPUT:

```
First column     5
Second column    6
Third column     4
dtype: int64
```

CODE:

```
#Part b

df.dropna(axis=1,thresh=45)
```

OUTPUT:

	First column	Third column
0	1.350895	-0.976909
1	-2.313891	-0.549008
2	-1.561012	0.003799
3	NaN	NaN
4	0.659472	0.414668
5	-0.193832	NaN
6	0.567887	0.949145
7	0.686564	-0.470725
8	0.260752	0.525147
9	0.580914	-0.134727
10	-0.214105	NaN
11	-0.709970	0.542781
12	-1.110686	NaN

36	NaN	-0.070108
37	-0.034891	-0.577948
38	1.505285	-0.944063
39	0.663310	-0.514068
40	0.741185	0.549174
41	0.382114	-0.332176
42	-0.731859	-0.485785
43	NaN	1.953285
44	-0.084395	1.936875
45	-0.386071	0.659375
46	0.554185	-1.285035
47	-0.073791	-0.759759
48	2.424553	1.230991
49	-0.755168	-1.116611

CODE:

```
#Part c

a=df.sum(axis=1).idxmax()

df.drop(index=a)

#print("Maximum Value", a)
```

OUTPUT:

	First column	Second column	Third column
0	-0.828871	-0.006589	-2.155317
1	1.352588	1.082039	-0.341240
2	NaN	1.498434	0.644644
3	-1.680507	-1.053472	-0.103793
4	NaN	0.373785	0.566714
5	1.106079	0.307390	-0.585303
6	-0.903279	-2.301988	-0.419412
7	-0.930783	0.601676	-0.736852
8	-0.437680	-1.202162	NaN
9	-0.718984	0.820854	1.705943
10	0.240469	-1.848375	NaN
11	0.159883	NaN	-0.791524
12	NaN	-0.362969	-0.854236

CODE:

```
#part d

df.sort_values('First column')
```

OUTPUT:

	First column	Second column	Third column
43	-2.060306	2.443549	-1.661335
35	-1.708430	1.064451	0.415539
3	-1.680507	-1.053472	-0.103793
47	-1.495339	NaN	-1.075393
25	-1.402827	0.224075	0.166391
40	-1.401055	-0.882812	-2.121886
17	-1.220660	-1.650634	0.456247
32	-1.094348	NaN	NaN
46	-0.970722	-0.046604	-2.195692
7	-0.930783	0.601676	-0.736852
6	-0.903279	-2.301988	-0.419412
26	-0.899136	1.295985	0.790716

CODE:

```
#part e

df.drop_duplicates('First column')
```

OUTPUT:

	First column	Second column	Third column
0	-0.828871	-0.006589	-2.155317
1	1.352588	1.082039	-0.341240
2	NaN	1.498434	0.644644
3	-1.680507	-1.053472	-0.103793
5	1.106079	0.307390	-0.585303
6	-0.903279	-2.301988	-0.419412
7	-0.930783	0.601676	-0.736852
8	-0.437680	-1.202162	NaN
9	-0.718984	0.820854	1.705943
10	0.240469	-1.848375	NaN
11	0.159883	NaN	-0.791524
13	0.239478	-0.409051	0.913603

CODE:

```
#Part e : Correlation

df['First column'].corr(df['Second column'])
```

OUTPUT:

```
-0.058767349805649814
```

CODE:

```
df['Second column'].cov(df['Third column'])
```

OUTPUT:

```
0.015499521829757204
```

CODE:

```
#Part g : Outlier detection

outlier=pd.Series(data=False,index=df.index)

for col in df.columns:


    Q1= df[col].quantile(0.25)

    Q3= df[col].quantile(0.75)

    IQR=Q3-Q1

    lower_bound = Q1-(1.5 * IQR)

    upper_bound = Q3+(1.5 * IQR)
```

```
    outlier |= (df[col] < lower_bound) | (df[col] > upper_bound)

df=df[~outlier]

print(df)
```

OUTPUT:

```
    First column  Second column  Third column
0      -0.828871      -0.006589     -2.155317
1       1.352588       1.082039     -0.341240
2            NaN       1.498434      0.644644
3      -1.680507      -1.053472     -0.103793
4            NaN       0.373785      0.566714
5       1.106079       0.307390     -0.585303
6      -0.903279      -2.301988     -0.419412
7      -0.930783       0.601676     -0.736852
8      -0.437680      -1.202162           NaN
9      -0.718984       0.820854      1.705943
10      0.240469      -1.848375           NaN
11      0.159883            NaN     -0.791524
12           NaN      -0.362969     -0.854236
13      0.239478      -0.409051      0.913603
14      0.848119       0.340057      0.484156
15      0.515030      -0.719418      1.820366
16      1.236557      -1.537588     -0.924539
17     -1.220660      -1.650634      0.456247
18     -0.831512       0.648471      0.307235
19     -0.604259      -0.711677     -1.097490
20           NaN            NaN           NaN
21      0.042324       1.034994     -0.485485
22     -0.022265       2.205114     -0.663128
```

CODE:

```
#part h

df['Second column']= pd.cut(df['Second column'],bins=5)

df['Second column']
```

OUTPUT:

```
0     (-1.883, -0.885]
1      (-0.885, 0.114]
2      (-0.885, 0.114]
3       (0.114, 1.113]
4       (1.113, 2.112]
5      (-0.885, 0.114]
6      (-0.885, 0.114]
7       (0.114, 1.113]
8       (0.114, 1.113]
9     (-2.887, -1.883]
10     (-0.885, 0.114]
11     (-0.885, 0.114]
12      (1.113, 2.112]
13      (0.114, 1.113]
14      (0.114, 1.113]
15      (0.114, 1.113]
16                 NaN
17     (-0.885, 0.114]
18     (-0.885, 0.114]
19                 NaN
20     (-0.885, 0.114]
21     (-0.885, 0.114]
```

```
34    (-1.883, -0.885]
35      (0.114, 1.113]
36                 NaN
37     (-0.885, 0.114]
38      (0.114, 1.113]
39      (0.114, 1.113]
40      (0.114, 1.113]
41      (0.114, 1.113]
42      (0.114, 1.113]
43     (-0.885, 0.114]
44      (0.114, 1.113]
45      (1.113, 2.112]
46      (0.114, 1.113]
47    (-2.887, -1.883]
48                 NaN
49      (1.113, 2.112]
Name: Second column, dtype: category
Categories (5, interval[float64, right]): [(-2.887, -1.883] < (-1.883, -0.885] < (-0.885, 0.114] <
                                           (0.114, 1.113] < (1.113, 2.112]]
```

Question4

CODE:

```
import numpy as np

import pandas as pd


from google.colab import drive

drive.mount('/content/drive')

#adding attendence files

df=pd.read_excel("/content/drive/MyDrive/Colab Notebooks/Attendence1.xlsx")

df1=pd.read_excel("/content/drive/MyDrive/Colab Notebooks/Attendence2.xlsx")

df
```

OUTPUT:

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Chetan	11:15:00	40
1	Chetna	11:00:00	50
2	Dhwani	11:02:00	30
3	Himanshi	11:00:00	40
4	konicca	11:10:00	50

CODE:

```
df1
```

OUTPUT:

[ ]

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Amisha	11:15:00	40
1	Ayush Jha	11:02:00	30
2	Chetna	11:00:00	50
3	Himanshi	11:00:00	40
4	khushi	11:10:00	50

CODE:

```
df.parse_dates = ("Time of joining")

df
```

OUTPUT:

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Chetan	11:15:00	40
1	Chetna	11:00:00	50
2	Dhwani	11:02:00	30
3	Himanshi	11:00:00	40
4	konicca	11:10:00	50

CODE:

```
df1.parse_dates = ("Time of joining")

df1
```

OUTPUT:

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Amisha	11:15:00	40
1	Ayush Jha	11:02:00	30
2	Chetna	11:00:00	50
3	Himanshi	11:00:00	40
4	khushi	11:10:00	50

CODE:

```
        #part a

df.merge(df1, how='inner', on='NAME')
```

OUTPUT:

	NAME	TIME OF JOINING_x	DURATION (IN MIN)_x	TIME OF JOINING_y	DURATION (IN MIN)_y
0	Chetna	11:00:00	50	11:00:00	50
1	Himanshi	11:00:00	40	11:00:00	40

CODE:

```
df
```

OUTPUT:

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Chetan	11:15:00	40
1	Chetna	11:00:00	50
2	Dhwani	11:02:00	30
3	Himanshi	11:00:00	40
4	konicca	11:10:00	50

CODE:

```
#part - b

df.merge(df1,how="outer")
```

OUTPUT:

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Chetan	11:15:00	40
1	Chetna	11:00:00	50
2	Dhwani	11:02:00	30
3	Himanshi	11:00:00	40
4	konicca	11:10:00	50
5	Amisha	11:15:00	40
6	Ayush Jha	11:02:00	30
7	khushi	11:10:00	50

CODE:

```
#part -c

a=pd.concat([df,df1],ignore_index=True)

a
```

OUTPUT:

]

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Chetan	11:15:00	40
1	Chetna	11:00:00	50
2	Dhwani	11:02:00	30
3	Himanshi	11:00:00	40
4	konicca	11:10:00	50
5	Amisha	11:15:00	40
6	Ayush Jha	11:02:00	30
7	Chetna	11:00:00	50
8	Himanshi	11:00:00	40
9	khushi	11:10:00	50

CODE:

```
len(a)
```

OUTPUT:

```
10
```

CODE:

```
#part -d

b=df.merge(df1,how="outer")

b
```

OUTPUT:

	NAME	TIME OF JOINING	DURATION (IN MIN)
0	Chetan	11:15:00	40
1	Chetna	11:00:00	50
2	Dhwani	11:02:00	30
3	Himanshi	11:00:00	40
4	konicca	11:10:00	50
5	Amisha	11:15:00	40
6	Ayush Jha	11:02:00	30
7	khushi	11:10:00	50

CODE:

```
c=b.set_index(keys=["NAME","DURATION (IN MIN)"])

c
```

OUTPUT:

		TIME OF JOINING
NAME	DURATION (IN MIN)	
Chetan	40	11:15:00
Chetna	50	11:00:00
Dhwani	30	11:02:00
Himanshi	40	11:00:00
konicca	50	11:10:00
Amisha	40	11:15:00
Ayush Jha	30	11:02:00
khushi	50	11:10:00

CODE:

```
c.describe()
```

OUTPUT:

	TIME OF JOINING
count	8
unique	4
top	11:15:00
freq	2

Question5

CODE:

```
from matplotlib import pyplot as plt

import seaborn

import pandas as pd

import numpy as np

from google.colab import drive

drive.mount('/content/drive')

file1 = pd.read_csv(r"/content/drive/MyDrive/Colab Notebooks/iris_csv.csv",header=None,)

file1
```

OUTPUT:

	0	1	2	3	4
0	sepallength	sepalwidth	petallength	petalwidth	class
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
...	...	...	...	...	...
146	6.7	3.0	5.2	2.3	Iris-virginica
147	6.3	2.5	5.0	1.9	Iris-virginica
148	6.5	3.0	5.2	2.0	Iris-virginica
149	6.2	3.4	5.4	2.3	Iris-virginica
150	5.9	3.0	5.1	1.8	Iris-virginica

151 rows × 5 columns

CODE:

```
file1.columns=["SepalLengthCm","SepalWidthCm" ,"PetalLengthCm","PetalWidthCm","Species"]

file1
```

OUTPUT:

[ ]

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	sepallength	sepalwidth	petallength	petalwidth	class
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
...	...	...	...	...	...
146	6.7	3.0	5.2	2.3	Iris-virginica
147	6.3	2.5	5.0	1.9	Iris-virginica
148	6.5	3.0	5.2	2.0	Iris-virginica
149	6.2	3.4	5.4	2.3	Iris-virginica
150	5.9	3.0	5.1	1.8	Iris-virginica

151 rows × 5 columns

CODE:

```
frequency=file1["Species"].value_counts()

frequency
```

OUTPUT:

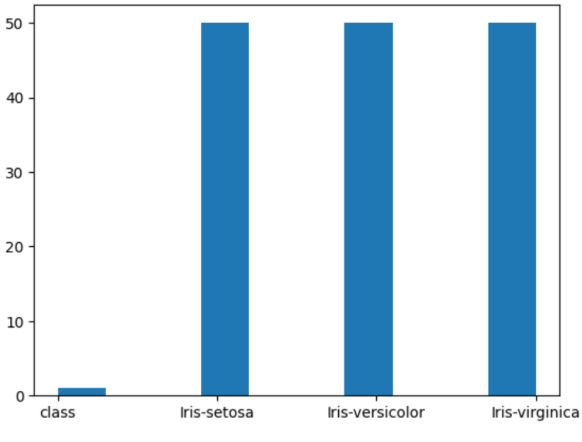
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
class               1
Name: Species, dtype: int64
```

CODE:

```
plt.hist(file1["Species"])
```

OUTPUT:

```
(array([ 1.,  0.,  0., 50.,  0.,  0., 50.,  0.,  0., 50.]),
 array([0. , 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.4, 2.7, 3. ]),
 <BarContainer object of 10 artists>)
```



CODE:

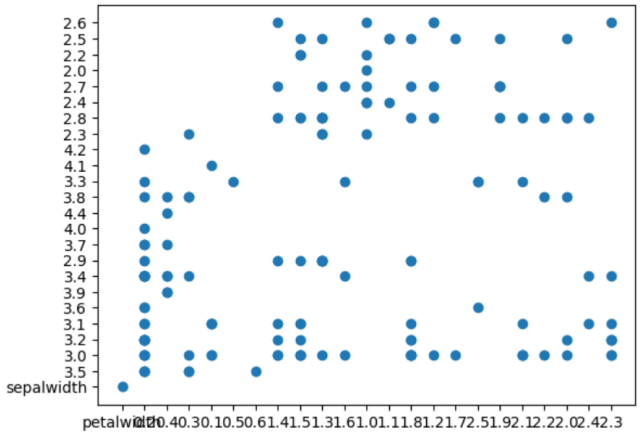
```
a=file1["PetalWidthCm"]

b=file1["SepalWidthCm"]

plt.scatter(a,b)
```

OUTPUT:

```
<matplotlib.collections.PathCollection at 0x7da5a6bb64a0>
```



CODE:

pip install scipy

OUTPUT:

```
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (1.11.4)
Requirement already satisfied: numpy<1.28.0,>=1.21.6 in /usr/local/lib/python3.10/dist-packages (from scipy) (1.23.5)
```

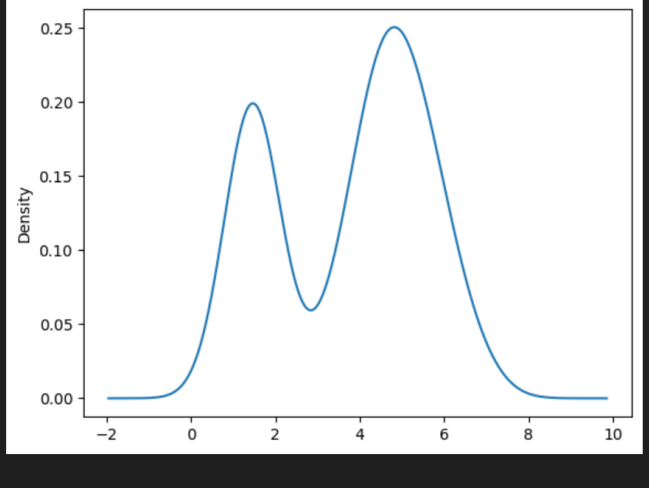
CODE:

x=file1["PetalLengthCm"]

x.plot.density()

OUTPUT:

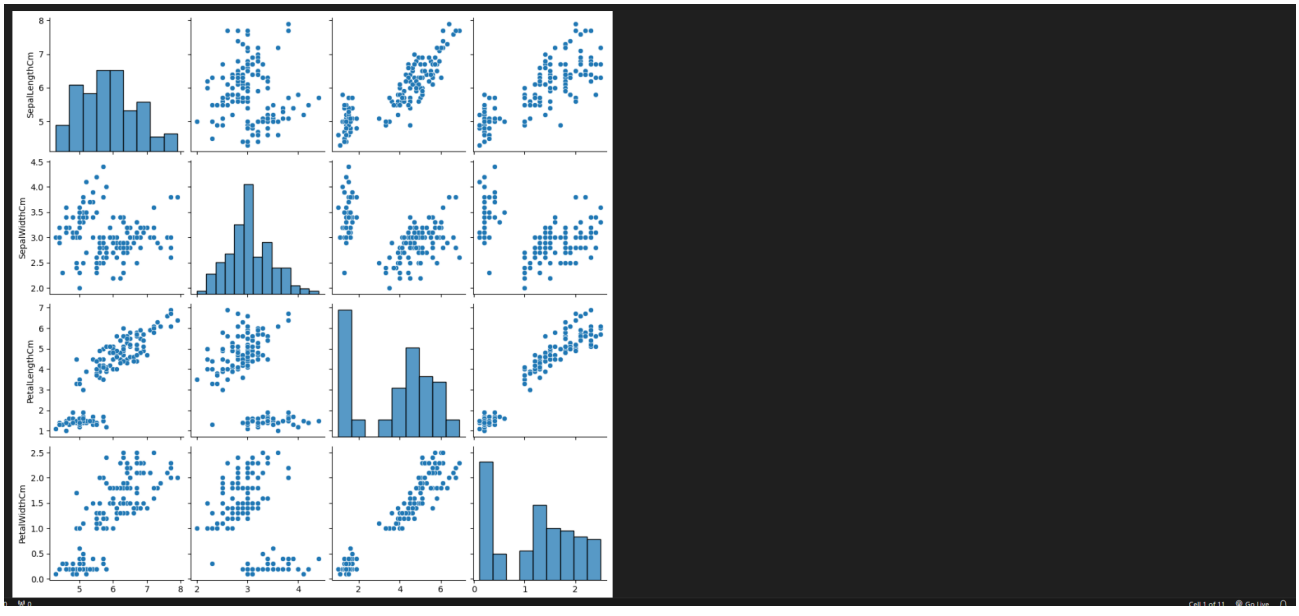
```
<Axes: ylabel='Density'>
```



CODE:

seaborn.pairplot(file1)

OUTPUT:



Question6

CODE:

```
import pandas as pd

from google.colab import drive

drive.mount('/content/drive')

data= pd.read_csv("/content/drive/MyDrive/Colab Notebooks/weather.csv")

data
```

OUTPUT:

	day	city	temperature	windspeed	event
0	1/1/2017	new york	32	6	Rain
1	1/2/2017	new york	36	7	Sunny
2	1/3/2017	new york	28	12	Snow
3	1/4/2017	new york	33	7	Sunny
4	1/1/2017	mumbai	90	5	Sunny
5	1/2/2017	mumbai	85	12	Fog
6	1/3/2017	mumbai	87	15	Fog
7	1/4/2017	mumbai	92	5	Rain
8	1/1/2017	paris	45	20	Sunny
9	1/2/2017	paris	50	13	Cloudy
10	1/3/2017	paris	54	8	Cloudy
11	1/4/2017	paris	42	10	Cloudy

CODE:

```
#part a
```

```
data.groupby('city')['temperature'].mean()
```

OUTPUT:

```
city
mumbai      88.50
new york    32.25
paris       47.75
Name: temperature, dtype: float64
```

CODE:

```
#part b

from datetime import datetime as dt

data['day']=pd.to_datetime(data['day']).dt.strftime('%d-%m-%y')

data
```

OUTPUT:

	day	city	temperature	windspeed	event
0	01-01-17	new york	32	6	Rain
1	02-01-17	new york	36	7	Sunny
2	03-01-17	new york	28	12	Snow
3	04-01-17	new york	33	7	Sunny
4	01-01-17	mumbai	90	5	Sunny
5	02-01-17	mumbai	85	12	Fog
6	03-01-17	mumbai	87	15	Fog
7	04-01-17	mumbai	92	5	Rain
8	01-01-17	paris	45	20	Sunny
9	02-01-17	paris	50	13	Cloudy
10	03-01-17	paris	54	8	Cloudy
11	04-01-17	paris	42	10	Cloudy

CODE:

```
data['day'].fillna(method='ffill')
```

OUTPUT:

```
0     1/1/2017
1     1/2/2017
2     1/3/2017
3     1/4/2017
4     1/1/2017
5     1/2/2017
6     1/3/2017
7     1/4/2017
8     1/1/2017
9     1/2/2017
10    1/3/2017
11    1/4/2017
Name: day, dtype: object
```

CODE:

```
data_agg= data.groupby(['city','event']).agg({'temperature':sum})

print(data_agg.sort_values(by='city',ascending=False))
```

OUTPUT:

```
                temperature
city     event
paris    Cloudy          146
         Sunny            45
new york Rain             32
         Snow             28
         Sunny            69
mumbai   Fog             172
         Rain             92
         Sunny            90
```

CODE:

```
weather=data.groupby(['event',pd.cut(df.windspeed,10)]).size()

weather
```

OUTPUT:

```
event   windspeed
Cloudy  (4.985, 6.5]    0
        (6.5, 8.0]      1
        (8.0, 9.5]      0
        (9.5, 11.0]     1
        (11.0, 12.5]    0
        (12.5, 14.0]    1
        (14.0, 15.5]    0
        (15.5, 17.0]    0
        (17.0, 18.5]    0
        (18.5, 20.0]    0
Fog     (4.985, 6.5]    0
        (6.5, 8.0]      0
        (8.0, 9.5]      0
        (9.5, 11.0]     0
        (11.0, 12.5]    1
        (12.5, 14.0]    0
        (14.0, 15.5]    1
        (15.5, 17.0]    0
        (17.0, 18.5]    0
        (18.5, 20.0]    0
Rain    (4.985, 6.5]    2
        (6.5, 8.0]      0
        (8.0, 9.5]      0
        (9.5, 11.0]     0
        (11.0, 12.5]    0
        (12.5, 14.0]    0
        (14.0, 15.5]    0
        (15.5, 17.0]    0
        (17.0, 18.5]    0
        (18.5, 20.0]    0
Snow    (4.985, 6.5]    0
        (6.5, 8.0]      0
        (8.0, 9.5]      0
        (9.5, 11.0]     0
        (11.0, 12.5]    1
        (12.5, 14.0]    0
        (14.0, 15.5]    0
        (15.5, 17.0]    0
        (17.0, 18.5]    0
        (18.5, 20.0]    0
Sunny   (4.985, 6.5]    1
        (6.5, 8.0]      2
        (8.0, 9.5]      0
        (9.5, 11.0]     0
        (11.0, 12.5]    0
        (12.5, 14.0]    0
        (14.0, 15.5]    0
        (15.5, 17.0]    0
        (17.0, 18.5]    0
        (18.5, 20.0]    1
dtype: int64
```

Question7

CODE:

```
import pandas as pd
```

```
import numpy as np

data={

   'Name' : ['Mudit Chauhan','Seema Chopra','Rani Gupta','Aditya Narayan','Sanjeev Sahni','Prakash Kumar','Ritu Aggarwal','Akshay Goel','Meeta Kulkarni','Preeti Ahuja','Sunil Das Gupta','Sonali Sapre','Rashmi Talwar','Ashish Dubey','Kiran Sharma','Sameer Bansal'],

   'Birth_Month' : ['Dec','Jan','Mar','Oct','Feb','Dec','Sep','Aug','Jul','Nov','Apr','Jan','Jun','May','Feb','Oct'],

   'Gender' : ['M','F','F','M','M','M','F','M','F','F','M','F','F','M','F','M'],

   'Pass_Division' : ['III','II','I','I','II','III','I','I','II','II','III','I','III','II','II','I']

}

df=pd.DataFrame(data)

df
```

OUTPUT:

	Name	Birth_Month	Gender	Pass_Division
0	Mudit Chauhan	Dec	M	III
1	Seema Chopra	Jan	F	II
2	Rani Gupta	Mar	F	I
3	Aditya Narayan	Oct	M	I
4	Sanjeev Sahni	Feb	M	II
5	Prakash Kumar	Dec	M	III
6	Ritu Aggarwal	Sep	F	I
7	Akshay Goel	Aug	M	I
8	Meeta Kulkarni	Jul	F	II
9	Preeti Ahuja	Nov	F	II
10	Sunil Das Gupta	Apr	M	III
11	Sonali Sapre	Jan	F	I
12	Rashmi Talwar	Jun	F	III
13	Ashish Dubey	May	M	II
14	Kiran Sharma	Feb	F	II

CODE:

```
                    #Part - A

df=pd.get_dummies(df,columns=['Gender','Pass_Division'])

df.info()
```

OUTPUT:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16 entries, 0 to 15
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Name               16 non-null     object
 1   Birth_Month        16 non-null     object
 2   Gender_F           16 non-null     uint8
 3   Gender_M           16 non-null     uint8
 4   Pass_Division_I    16 non-null     uint8
 5   Pass_Division_II   16 non-null     uint8
 6   Pass_Division_III  16 non-null     uint8
dtypes: object(2), uint8(5)
memory usage: 464.0+ bytes
```

CODE:

df

OUTPUT:

	Name	Birth_Month	Gender_F	Gender_M	Pass_Division_I	Pass_Division_II	Pass_Division_III
1	Seema Chopra	Jan	1	0	0	1	0
11	Sonali Sapre	Jan	1	0	1	0	0
4	Sanjeev Sahni	Feb	0	1	0	1	0
14	Kiran Sharma	Feb	1	0	0	1	0
2	Rani Gupta	Mar	1	0	1	0	0
10	Sunil Das Gupta	Apr	0	1	0	0	1
13	Ashish Dubey	May	0	1	0	1	0
12	Rashmi Talwar	Jun	1	0	0	0	1
8	Meeta Kulkarni	Jul	1	0	0	1	0
7	Akshay Goel	Aug	0	1	1	0	0
6	Ritu Aggarwal	Sep	1	0	1	0	0
3	Aditya Narayan	Oct	0	1	1	0	0
15	Sameer Bansal	Oct	0	1	1	0	0
9	Preeti Ahuja	Nov	1	0	0	1	0
0	Mudit Chauhan	Dec	0	1	0	0	1

CODE:

```
#Part-B
```

```
df["Birth_Month"]=df.Birth_Month.astype("category")
```

df

OUTPUT:

[ ]

	Name	Birth_Month	Gender_F	Gender_M	Pass_Division_I	Pass_Division_II	Pass_Division_III
0	Mudit Chauhan	Dec	0	1	0	0	1
1	Seema Chopra	Jan	1	0	0	1	0
2	Rani Gupta	Mar	1	0	1	0	0
3	Aditya Narayan	Oct	0	1	1	0	0
4	Sanjeev Sahni	Feb	0	1	0	1	0
5	Prakash Kumar	Dec	0	1	0	0	1
6	Ritu Aggarwal	Sep	1	0	1	0	0
7	Akshay Goel	Aug	0	1	1	0	0
8	Meeta Kulkarni	Jul	1	0	0	1	0
9	Preeti Ahuja	Nov	1	0	0	1	0
10	Sunil Das Gupta	Apr	0	1	0	0	1
11	Sonali Sapre	Jan	1	0	1	0	0
12	Rashmi Talwar	Jun	1	0	0	0	1
13	Ashish Dubey	May	0	1	0	1	0
14	Kiran Sharma	Feb	1	0	0	1	0
15	Sameer Bansal	Oct	0	1	1	0	0

CODE:

df.dtypes

OUTPUT:

```
Name                  object
Birth_Month         category
Gender_F               uint8
Gender_M               uint8
Pass_Division_I        uint8
Pass_Division_II       uint8
Pass_Division_III      uint8
dtype: object
```

CODE:

```
month=['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']

df['Birth_Month']=pd.Categorical(df['Birth_Month'] , categories = month, order='true')    #used for passing the list for sort values

df.sort_values(by='Birth_Month' ,inplace=True)

df
```

OUTPUT:

[ ]

	Name	Birth_Month	Gender_F	Gender_M	Pass_Division_I	Pass_Division_II	Pass_Division_III
1	Seema Chopra	Jan	1	0	0	1	0
11	Sonali Sapre	Jan	1	0	1	0	0
4	Sanjeev Sahni	Feb	0	1	0	1	0
14	Kiran Sharma	Feb	1	0	0	1	0
2	Rani Gupta	Mar	1	0	1	0	0
10	Sunil Das Gupta	Apr	0	1	0	0	1
13	Ashish Dubey	May	0	1	0	1	0
12	Rashmi Talwar	Jun	1	0	0	0	1
8	Meeta Kulkarni	Jul	1	0	0	1	0
7	Akshay Goel	Aug	0	1	1	0	0
6	Ritu Aggarwal	Sep	1	0	1	0	0
3	Aditya Narayan	Oct	0	1	1	0	0
15	Sameer Bansal	Oct	0	1	1	0	0
9	Preeti Ahuja	Nov	1	0	0	1	0
0	Mudit Chauhan	Dec	0	1	0	0	1

Question8

CODE:

```
import pandas as pd

from google.colab import drive

drive.mount('/content/drive')

data=pd.read_excel("/content/drive/MyDrive/Colab Notebooks/Ques_8.xlsx")

data
```

OUTPUT:

	Name	Gender	MonthlyIncome(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	Vats	Male	71900

CODE:

```
df=pd.DataFrame(data)

df
```

OUTPUT:

	Name	Gender	MonthlyIncome(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	Vats	Male	71900

CODE:

```
family_gross_income = df.groupby('Name')['MonthlyIncome(Rs.)'].sum().reset_index()

print(family_gross_income)
```

OUTPUT:

```
    Name  MonthlyIncome(Rs.)
0  Kumar              253530
1   Shah              281400
2   Vats              335050
```

CODE:

```
maxIncomer = df.loc[df.groupby('Name')['MonthlyIncome(Rs.)'].idxmax()]

print(maxIncomer[['Name', 'Gender', 'MonthlyIncome(Rs.)']])
```

OUTPUT:

```
    Name  Gender  MonthlyIncome(Rs.)
5  Kumar    Male              103000
0   Shah    Male              114000
4   Vats  Female              155000
```

CODE:

```
inc = df[df['MonthlyIncome(Rs.)'] > 60000.00]

print(inc)
```

OUTPUT:

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

CODE:

```
femaleMem = df[(df['Name'] == 'Shah') & (df['Gender'] == 'Female')]
```

```
avgInc = femaleMem['MonthlyIncome(Rs.)'].mean()

print(avgInc)
```

OUTPUT:

```
112400.0
```