NAME - DEVKI NANDAN

ROLL NO - 10909

COURSE - BSC(H) COMPUTER SCIENCE ¶

SUBJECT - DATA ANALYSIS AND VISUALIZATION

EXAMINATION ROLL NO - 21035570026

In [1]:
```
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt
import seaborn
import datetime as dt
```

Question 1

In [2]:
```
dict1={'Boys': [72, 68, 70, 69, 74], 'Girls': [63, 65, 69, 62, 61]}
```

In [3]:
```
list1=[{'Boys':i,'Girls':j}for i,j in zip(dict1['Boys'],dict1['Girls'])]
```

In [4]:
```
list1
```

Out[4]:
```
[{'Boys': 72, 'Girls': 63},
 {'Boys': 68, 'Girls': 65},
 {'Boys': 70, 'Girls': 69},
 {'Boys': 69, 'Girls': 62},
 {'Boys': 74, 'Girls': 61}]
```

In [ ]:
```

```

Question 2

In [5]:
```
array = np.random.randint(1,100,size=(3,4))
array
```

Out[5]:
```
array([[25, 25, 82,  9],
       [87, 37, 63, 95],
       [ 4, 26, 75, 25]])
```

part A

In [6]:
```
np.mean(array,axis=1)
```

Out[6]:
```
array([35.25, 70.5 , 32.5 ])
```

In [7]:
```
array.std(axis=1)
```

Out[7]:
```
array([27.77026287, 22.64398375, 26.06242506])
```

In [8]:
```
array.var(axis=1)
```

Out[8]:
```
array([771.1875, 512.75  , 679.25  ])
```

part B

In [9]:
```
B=[56,48,22,41,78,91,24,46,8,33]
a=np.argsort(B)
a
```

Out[9]:
```
array([8, 2, 6, 9, 3, 7, 1, 0, 4, 5], dtype=int64)
```

part C

In [10]:
```
m=int(input("Enter first dimension"))
n=int(input("Enter second dimension"))
df = np.random.randint(0,10,size=(m,n))
df
```

```
Enter first dimension3
Enter second dimension4
```

Out[10]:
```
array([[8, 7, 7, 5],
       [0, 8, 6, 4],
       [5, 4, 9, 3]])
```

In [11]:
```
df.shape
```

Out[11]:
```
(3, 4)
```

In [12]:
```
df.dtype
```

Out[12]:
```
dtype('int32')
```

In [13]:
```
type(df)
```

Out[13]:
```
numpy.ndarray
```

In [14]:
```
df.reshape(n,m)
```

Out[14]:
```
array([[8, 7, 7],
       [5, 0, 8],
       [6, 4, 5],
       [4, 9, 3]])
```

part D

In [15]:
```
df=np.array([5,8,9,np.nan,3,4,6,0,1,0,1,0,0,np.nan])
zero=np.where(df==0)
zero
```

Out[15]:
```
(array([ 7,  9, 11, 12], dtype=int64),)
```

In [16]:
```
temp=np.nan
non_zero=np.where(df!=0)
non_zero
```

Out[16]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  8, 10, 13], dtype=int64),)
```

In [17]:
```
nan_val=np.isnan(df)
print(np.where(nan_val==True))
```

```
(array([ 3, 13], dtype=int64),)
```

In [ ]:
```

```

Question 3

In [18]:

```
data=np.random.randint(1,100,size=(50,3))
df=pd.DataFrame(data,columns=["First column","Second column","Third column"])

for i in range(15):
    r=np.random.randint(0,50)
    c=np.random.randint(0,3)
    df.iloc[r,c]=np.nan
df
```

Out[18]:

	First column	Second column	Third column
0	19.0	8.0	76.0
1	NaN	99.0	98.0
2	40.0	55.0	66.0
3	18.0	NaN	11.0
4	95.0	30.0	96.0
5	87.0	3.0	61.0
6	7.0	73.0	4.0
7	3.0	21.0	45.0
8	42.0	23.0	NaN
9	73.0	97.0	50.0
10	9.0	17.0	91.0
11	51.0	96.0	94.0
12	19.0	30.0	81.0
13	42.0	51.0	14.0
14	77.0	79.0	48.0
15	26.0	71.0	13.0
16	69.0	56.0	1.0
17	90.0	NaN	5.0
18	53.0	55.0	60.0
19	58.0	84.0	38.0
20	15.0	67.0	99.0
21	NaN	77.0	68.0
22	82.0	35.0	30.0
23	8.0	34.0	18.0
24	27.0	79.0	21.0

	First column	Second column	Third column
25	64.0	58.0	26.0
26	81.0	NaN	1.0
27	79.0	84.0	34.0
28	41.0	33.0	22.0
29	29.0	70.0	77.0
30	56.0	33.0	64.0
31	93.0	NaN	39.0
32	57.0	62.0	66.0
33	26.0	85.0	86.0
34	89.0	56.0	13.0
35	5.0	72.0	45.0
36	76.0	32.0	88.0
37	20.0	79.0	31.0
38	23.0	37.0	4.0
39	90.0	1.0	NaN
40	70.0	41.0	20.0
41	81.0	30.0	NaN
42	59.0	96.0	31.0
43	NaN	79.0	NaN
44	67.0	55.0	82.0
45	91.0	37.0	2.0
46	51.0	1.0	NaN
47	NaN	13.0	10.0
48	61.0	1.0	94.0
49	NaN	96.0	39.0

part A

In [19]:
```
df1=df.isnull().sum()
df1
```

Out[19]:
```
First column     5
Second column    4
Third column     5
dtype: int64
```

part B

In [20]:
```
df.dropna(axis=1,thresh=45)
```

Out[20]:

	First column	Second column	Third column
0	19.0	8.0	76.0
1	NaN	99.0	98.0
2	40.0	55.0	66.0
3	18.0	NaN	11.0
4	95.0	30.0	96.0
5	87.0	3.0	61.0
6	7.0	73.0	4.0
7	3.0	21.0	45.0
8	42.0	23.0	NaN
9	73.0	97.0	50.0
10	9.0	17.0	91.0
11	51.0	96.0	94.0
12	19.0	30.0	81.0
13	42.0	51.0	14.0
14	77.0	79.0	48.0
15	26.0	71.0	13.0
16	69.0	56.0	1.0
17	90.0	NaN	5.0
18	53.0	55.0	60.0
19	58.0	84.0	38.0
20	15.0	67.0	99.0
21	NaN	77.0	68.0
22	82.0	35.0	30.0
23	8.0	34.0	18.0
24	27.0	79.0	21.0

	First column	Second column	Third column
25	64.0	58.0	26.0
26	81.0	NaN	1.0
27	79.0	84.0	34.0
28	41.0	33.0	22.0
29	29.0	70.0	77.0
30	56.0	33.0	64.0
31	93.0	NaN	39.0
32	57.0	62.0	66.0
33	26.0	85.0	86.0
34	89.0	56.0	13.0
35	5.0	72.0	45.0
36	76.0	32.0	88.0
37	20.0	79.0	31.0
38	23.0	37.0	4.0
39	90.0	1.0	NaN
40	70.0	41.0	20.0
41	81.0	30.0	NaN
42	59.0	96.0	31.0
43	NaN	79.0	NaN
44	67.0	55.0	82.0
45	91.0	37.0	2.0
46	51.0	1.0	NaN
47	NaN	13.0	10.0
48	61.0	1.0	94.0
49	NaN	96.0	39.0

part C

In [21]:
```
a=df.sum(axis=1).idxmax()   #delete the row which has max sum
df.drop(index=a)
```

Out[21]:

	First column	Second column	Third column
0	19.0	8.0	76.0
1	NaN	99.0	98.0
2	40.0	55.0	66.0
3	18.0	NaN	11.0
4	95.0	30.0	96.0
5	87.0	3.0	61.0
6	7.0	73.0	4.0
7	3.0	21.0	45.0
8	42.0	23.0	NaN
9	73.0	97.0	50.0
10	9.0	17.0	91.0
12	19.0	30.0	81.0
13	42.0	51.0	14.0
14	77.0	79.0	48.0
15	26.0	71.0	13.0
16	69.0	56.0	1.0
17	90.0	NaN	5.0
18	53.0	55.0	60.0
19	58.0	84.0	38.0
20	15.0	67.0	99.0
21	NaN	77.0	68.0
22	82.0	35.0	30.0
23	8.0	34.0	18.0
24	27.0	79.0	21.0
25	64.0	58.0	26.0

	First column	Second column	Third column
26	81.0	NaN	1.0
27	79.0	84.0	34.0
28	41.0	33.0	22.0
29	29.0	70.0	77.0
30	56.0	33.0	64.0
31	93.0	NaN	39.0
32	57.0	62.0	66.0
33	26.0	85.0	86.0
34	89.0	56.0	13.0
35	5.0	72.0	45.0
36	76.0	32.0	88.0
37	20.0	79.0	31.0
38	23.0	37.0	4.0
39	90.0	1.0	NaN
40	70.0	41.0	20.0
41	81.0	30.0	NaN
42	59.0	96.0	31.0
43	NaN	79.0	NaN
44	67.0	55.0	82.0
45	91.0	37.0	2.0
46	51.0	1.0	NaN
47	NaN	13.0	10.0
48	61.0	1.0	94.0
49	NaN	96.0	39.0

part D

In [22]:
```
df.sort_values('First column')
```

Out[22]:

	First column	Second column	Third column
7	3.0	21.0	45.0
35	5.0	72.0	45.0
6	7.0	73.0	4.0
23	8.0	34.0	18.0
10	9.0	17.0	91.0
20	15.0	67.0	99.0
3	18.0	NaN	11.0
0	19.0	8.0	76.0
12	19.0	30.0	81.0
37	20.0	79.0	31.0
38	23.0	37.0	4.0
33	26.0	85.0	86.0
15	26.0	71.0	13.0
24	27.0	79.0	21.0
29	29.0	70.0	77.0
2	40.0	55.0	66.0
28	41.0	33.0	22.0
13	42.0	51.0	14.0
8	42.0	23.0	NaN
11	51.0	96.0	94.0
46	51.0	1.0	NaN
18	53.0	55.0	60.0
30	56.0	33.0	64.0
32	57.0	62.0	66.0
19	58.0	84.0	38.0

	First column	Second column	Third column
42	59.0	96.0	31.0
48	61.0	1.0	94.0
25	64.0	58.0	26.0
44	67.0	55.0	82.0
16	69.0	56.0	1.0
40	70.0	41.0	20.0
9	73.0	97.0	50.0
36	76.0	32.0	88.0
14	77.0	79.0	48.0
27	79.0	84.0	34.0
26	81.0	NaN	1.0
41	81.0	30.0	NaN
22	82.0	35.0	30.0
5	87.0	3.0	61.0
34	89.0	56.0	13.0
17	90.0	NaN	5.0
39	90.0	1.0	NaN
45	91.0	37.0	2.0
31	93.0	NaN	39.0
4	95.0	30.0	96.0
1	NaN	99.0	98.0
21	NaN	77.0	68.0
43	NaN	79.0	NaN
47	NaN	13.0	10.0
49	NaN	96.0	39.0

part E

In [23]:
```
df.drop_duplicates('First column')
```

Out[23]:

	First column	Second column	Third column
0	19.0	8.0	76.0
1	NaN	99.0	98.0
2	40.0	55.0	66.0
3	18.0	NaN	11.0
4	95.0	30.0	96.0
5	87.0	3.0	61.0
6	7.0	73.0	4.0
7	3.0	21.0	45.0
8	42.0	23.0	NaN
9	73.0	97.0	50.0
10	9.0	17.0	91.0
11	51.0	96.0	94.0
14	77.0	79.0	48.0
15	26.0	71.0	13.0
16	69.0	56.0	1.0
17	90.0	NaN	5.0
18	53.0	55.0	60.0
19	58.0	84.0	38.0
20	15.0	67.0	99.0
22	82.0	35.0	30.0
23	8.0	34.0	18.0
24	27.0	79.0	21.0
25	64.0	58.0	26.0
26	81.0	NaN	1.0
27	79.0	84.0	34.0

	First column	Second column	Third column
28	41.0	33.0	22.0
29	29.0	70.0	77.0
30	56.0	33.0	64.0
31	93.0	NaN	39.0
32	57.0	62.0	66.0
34	89.0	56.0	13.0
35	5.0	72.0	45.0
36	76.0	32.0	88.0
37	20.0	79.0	31.0
38	23.0	37.0	4.0
40	70.0	41.0	20.0
42	59.0	96.0	31.0
44	67.0	55.0	82.0
45	91.0	37.0	2.0
48	61.0	1.0	94.0

part F (1)

In [24]:
```
df['First column'].corr(df['Second column'])
```

Out[24]:
```
-0.10311976850010383
```

part F (2)

In [25]:
```
df['Second column'].cov(df['Third column'])
```

Out[25]:
```
-36.17134146341465
```

part G

In [26]:
```
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)
```

	First column	Second column	Third column
0	19.0	8.0	76.0
1	NaN	99.0	98.0
2	40.0	55.0	66.0
3	18.0	NaN	11.0
4	95.0	30.0	96.0
5	87.0	3.0	61.0
6	7.0	73.0	4.0
7	3.0	21.0	45.0
8	42.0	23.0	NaN
9	73.0	97.0	50.0
10	9.0	17.0	91.0
11	51.0	96.0	94.0
12	19.0	30.0	81.0
13	42.0	51.0	14.0
14	77.0	79.0	48.0
15	26.0	71.0	13.0
16	69.0	56.0	1.0
17	90.0	NaN	5.0
18	53.0	55.0	60.0
19	58.0	84.0	38.0
20	15.0	67.0	99.0
21	NaN	77.0	68.0
22	82.0	35.0	30.0
23	8.0	34.0	18.0
24	27.0	79.0	21.0
25	64.0	58.0	26.0
26	81.0	NaN	1.0
27	79.0	84.0	34.0
28	41.0	33.0	22.0
29	29.0	70.0	77.0
30	56.0	33.0	64.0
31	93.0	NaN	39.0
32	57.0	62.0	66.0
33	26.0	85.0	86.0
34	89.0	56.0	13.0
35	5.0	72.0	45.0
36	76.0	32.0	88.0
37	20.0	79.0	31.0
38	23.0	37.0	4.0
39	90.0	1.0	NaN

40	70.0	41.0	20.0
41	81.0	30.0	NaN
42	59.0	96.0	31.0
43	NaN	79.0	NaN
44	67.0	55.0	82.0
45	91.0	37.0	2.0
46	51.0	1.0	NaN
47	NaN	13.0	10.0
48	61.0	1.0	94.0
49	NaN	96.0	39.0

part H

In [27]:
```
df['Second column']= pd.cut(df['Second column'],bins=5)
df['Second column']
```

Out[27]:
```
0     (0.902, 20.6]
1      (79.4, 99.0]
2      (40.2, 59.8]
3               NaN
4      (20.6, 40.2]
5     (0.902, 20.6]
6      (59.8, 79.4]
7      (20.6, 40.2]
8      (20.6, 40.2]
9      (79.4, 99.0]
10    (0.902, 20.6]
11     (79.4, 99.0]
12     (20.6, 40.2]
13     (40.2, 59.8]
14     (59.8, 79.4]
15     (59.8, 79.4]
16     (40.2, 59.8]
17              NaN
18     (40.2, 59.8]
19     (79.4, 99.0]
20     (59.8, 79.4]
21     (59.8, 79.4]
22     (20.6, 40.2]
23     (20.6, 40.2]
24     (59.8, 79.4]
25     (40.2, 59.8]
26              NaN
27     (79.4, 99.0]
28     (20.6, 40.2]
29     (59.8, 79.4]
30     (20.6, 40.2]
31              NaN
32     (59.8, 79.4]
33     (79.4, 99.0]
34     (40.2, 59.8]
35     (59.8, 79.4]
36     (20.6, 40.2]
37     (59.8, 79.4]
38     (20.6, 40.2]
39    (0.902, 20.6]
40     (40.2, 59.8]
```

```
41     (20.6, 40.2]
42     (79.4, 99.0]
43     (59.8, 79.4]
44     (40.2, 59.8]
45     (20.6, 40.2]
46    (0.902, 20.6]
47    (0.902, 20.6]
48    (0.902, 20.6]
49     (79.4, 99.0]
Name: Second column, dtype: category
Categories (5, interval[float64, right]): [(0.902, 20.6] < (20.6, 40.2] < (40.2, 59.8] < (59.8, 79.4] < (79.4, 99.0]]
```

In [ ]:

Question 4

In [28]:
```
df=pd.read_excel("day1.xlsx")
df1=pd.read_excel("day2.xlsx")
```

In [29]:
```
df
```

Out[29]:

	Name	Time of joining	Duration
0	Alice	09:15:00	50
1	Bob	09:30:00	40
2	Carol	09:45:00	30
3	Dave	10:00:00	50
4	Eve	10:15:00	40

In [30]:
```
df1
```

Out[30]:

	Name	Time of joining	Duration
0	Alice	09:00:00	40
1	Bob	09:15:00	50
2	Carol	09:30:00	30
3	David	10:45:00	40
4	Frank	10:00:00	50

In [31]:
```
df.parse_dates = ("Time of joining")
df
```

Out[31]:

	Name	Time of joining	Duration
0	Alice	09:15:00	50
1	Bob	09:30:00	40
2	Carol	09:45:00	30
3	Dave	10:00:00	50
4	Eve	10:15:00	40

In [32]:
```
df1.parse_dates=("Time of joining")
df1
```

Out[32]:

	Name	Time of joining	Duration
0	Alice	09:00:00	40
1	Bob	09:15:00	50
2	Carol	09:30:00	30
3	David	10:45:00	40
4	Frank	10:00:00	50

part A

In [33]:
```
df.merge(df1,how="inner",on="Name")
```

Out[33]:

	Name	Time of joining_x	Duration_x	Time of joining_y	Duration_y
0	Alice	09:15:00	50	09:00:00	40
1	Bob	09:30:00	40	09:15:00	50
2	Carol	09:45:00	30	09:30:00	30

part B

In [34]:
```
df.merge(df1,how="outer")
```

Out[34]:

	Name	Time of joining	Duration
0	Alice	09:15:00	50
1	Bob	09:30:00	40
2	Carol	09:45:00	30
3	Dave	10:00:00	50
4	Eve	10:15:00	40
5	Alice	09:00:00	40
6	Bob	09:15:00	50
7	Carol	09:30:00	30
8	David	10:45:00	40
9	Frank	10:00:00	50

part C

In [35]:
```
a=pd.concat([df,df1],ignore_index=True)
len(a)
```

Out[35]:
```
10
```

part D

In [36]:
```
b=df.merge(df1,how="outer")
b
```

Out[36]:

	Name	Time of joining	Duration
0	Alice	09:15:00	50
1	Bob	09:30:00	40
2	Carol	09:45:00	30
3	Dave	10:00:00	50
4	Eve	10:15:00	40
5	Alice	09:00:00	40
6	Bob	09:15:00	50
7	Carol	09:30:00	30
8	David	10:45:00	40
9	Frank	10:00:00	50

In [37]:
```
c=b.set_index(keys=["Name","Duration"])
c
```

Out[37]:

		Time of joining
Name	**Duration**	
Alice	**50**	09:15:00
Bob	**40**	09:30:00
Carol	**30**	09:45:00
Dave	**50**	10:00:00
Eve	**40**	10:15:00
Alice	**40**	09:00:00
Bob	**50**	09:15:00
Carol	**30**	09:30:00
David	**40**	10:45:00
Frank	**50**	10:00:00

In [38]:
```
c.describe()
```

Out[38]:

	Time of joining
count	10
unique	7
top	09:15:00
freq	2

In [ ]:
```

```

Question 5

In [39]:
```
file1 = pd.read_csv("iris.data",header=None,)
file1
```

Out[39]:

	0	1	2	3	4
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [40]:
```
file1.columns=["SepalLengthCm","SepalWidthCm" ,"PetalLengthCm","PetalWidthCm","Species"]
file1
```

Out[40]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

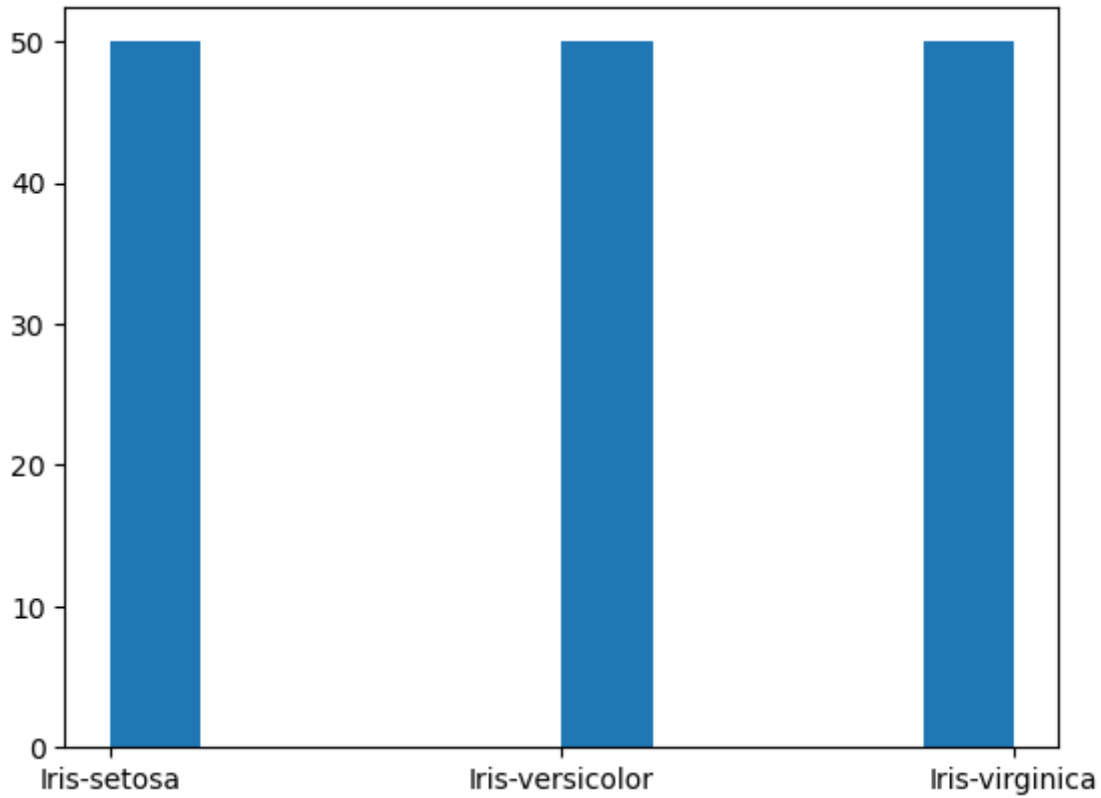
In [41]:
```
frequency=file1["Species"].value_counts()
frequency
```

Out[41]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

part A

In [42]:
```
plt.hist(file1["Species"])
```

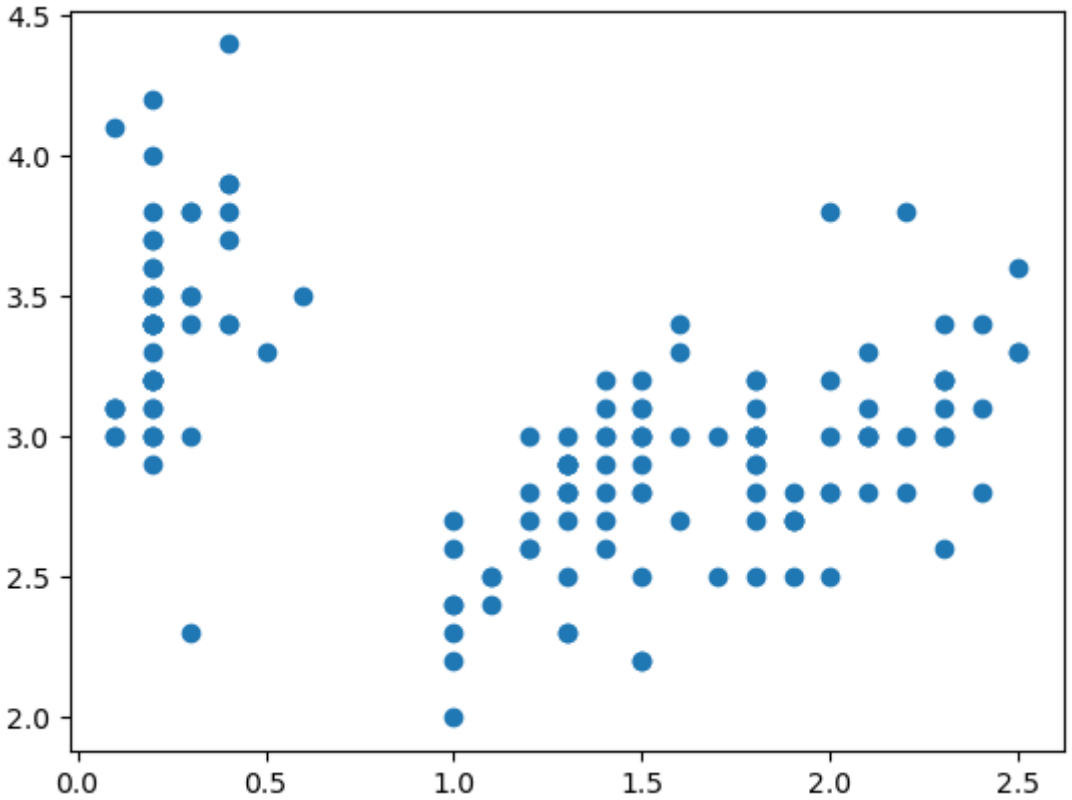
Out[42]:
```
(array([50.,  0.,  0.,  0.,  0., 50.,  0.,  0.,  0., 50.]),
 array([0. , 0.2, 0.4, 0.6, 0.8, 1. , 1.2, 1.4, 1.6, 1.8, 2. ]),
 <BarContainer object of 10 artists>)
```



part B

In [43]:
```
a=file1["PetalWidthCm"]
b=file1["SepalWidthCm"]
plt.scatter(a,b)
```

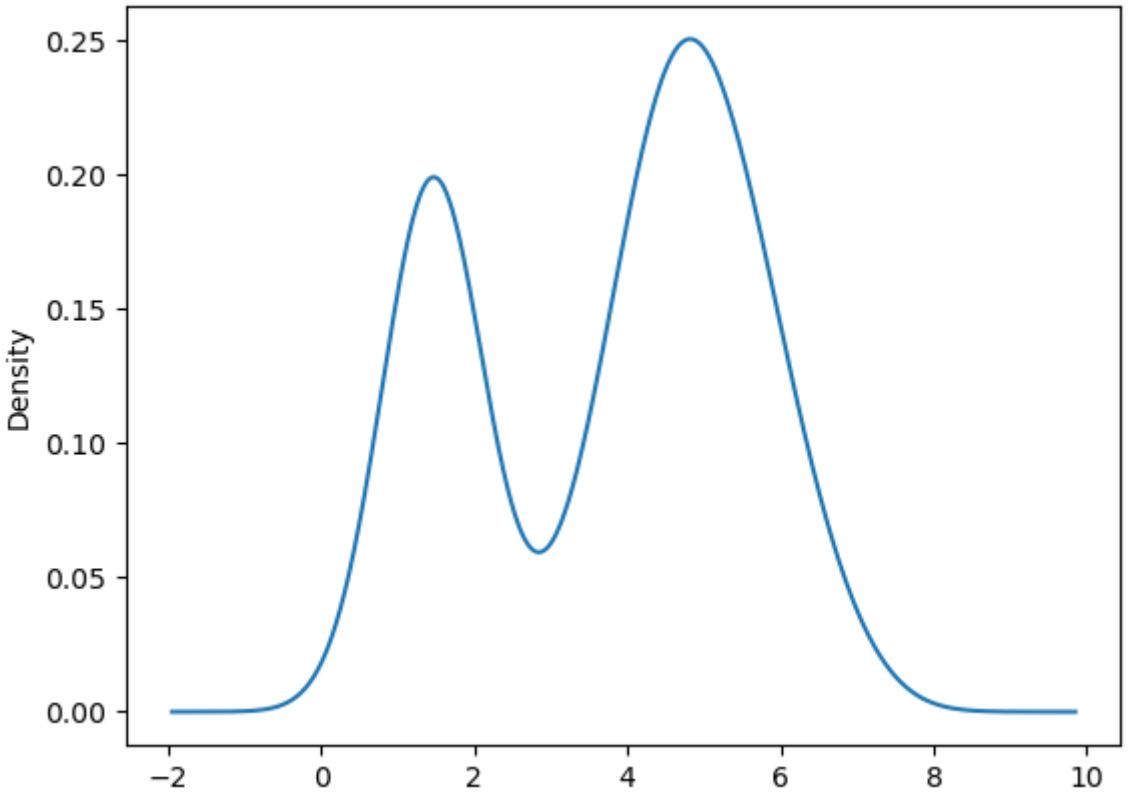
Out[43]: <matplotlib.collections.PathCollection at 0x1153e3542d0>



part C

In [44]:
```
x=file1["PetalLengthCm"]
x.plot.density()
```

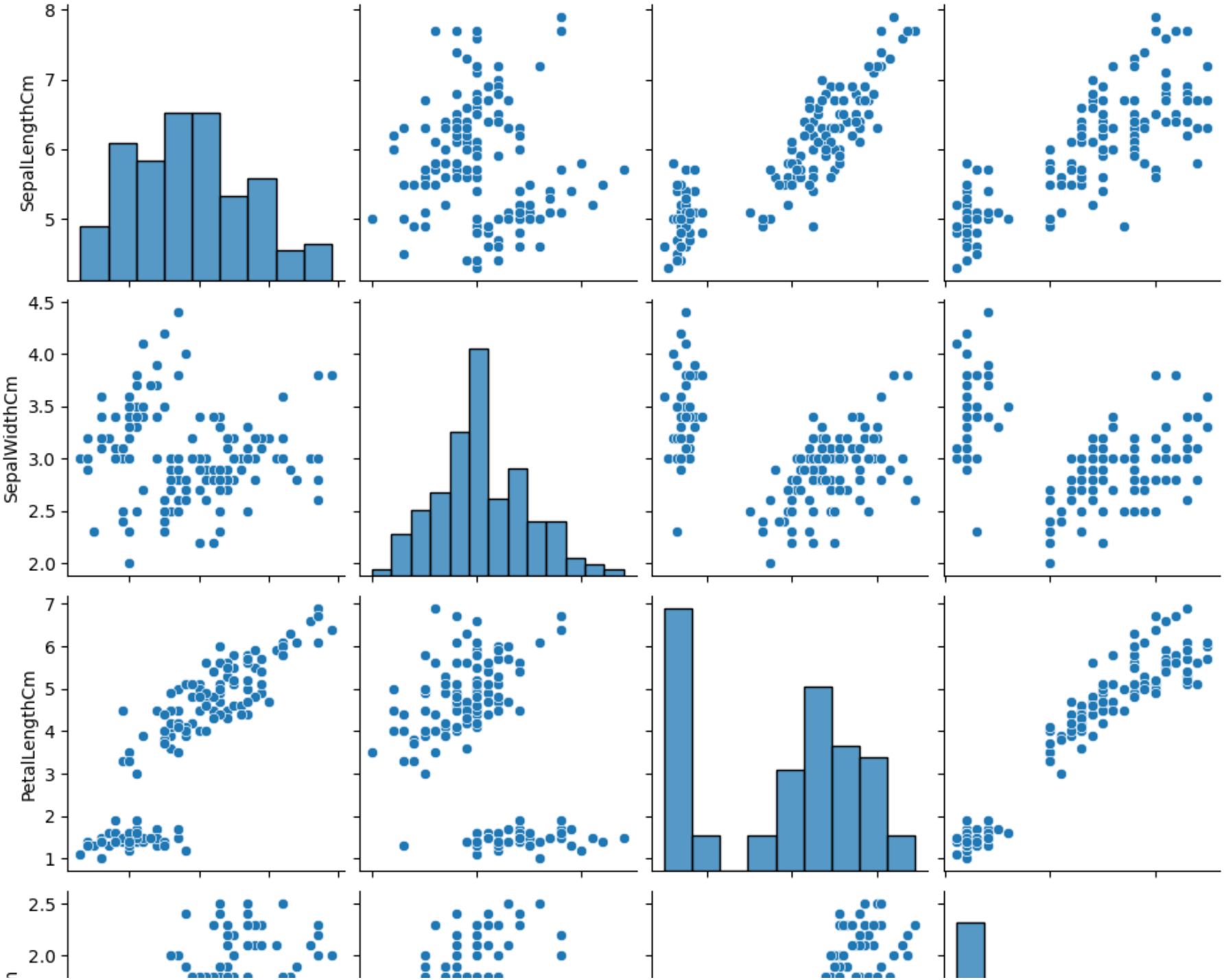
Out[44]: <Axes: ylabel='Density'>

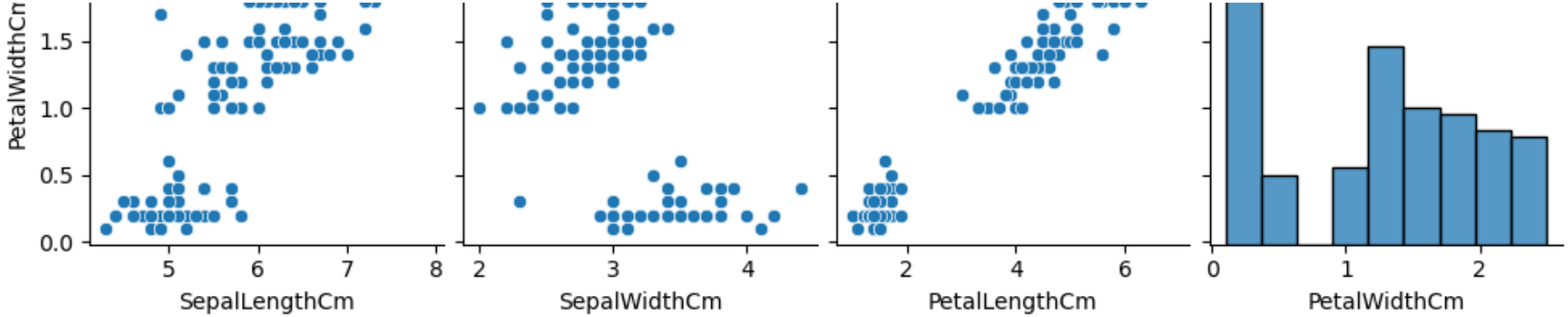


part D

In [45]:
```
seaborn.pairplot(file1)
```

Out[45]:
```
<seaborn.axisgrid.PairGrid at 0x1153e596bd0>
```


SepalLengthCm
SepalWidthCm
PetalLengthCm
8
7
6
5
4.5
4.0
3.5
3.0
2.5
2.0
7
6
5
4
3
2
1
2.5
2.0



In [ ]:

Question 6

In [46]:
```
df = pd.read_csv('weather_by_cities.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12 entries, 0 to 11
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   day          12 non-null     object
 1   city         12 non-null     object
 2   temperature  12 non-null     int64 
 3   windspeed    12 non-null     int64 
 4   event        12 non-null     object
dtypes: int64(2), object(3)
memory usage: 612.0+ bytes
```

part A

In [47]:
```
df.groupby('city')['temperature'].mean()
```

Out[47]:
```
city
mumbai      88.50
new york    32.25
paris       47.75
Name: temperature, dtype: float64
```

part B

In [48]:
```
df["day"].fillna(method = "ffill")
```

Out[48]:
```
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
```

part C

In [49]:
```
df['day'] = pd.to_datetime(df['day']).dt.strftime('%d-%m-%y')
df.head()
```

Out[49]:

	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

part D

In [50]:
```
df_agg = df.groupby(['event','city']).agg({'temperature' : sum})
print(df_agg.sort_values(by='event', ascending=False))
```

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

part E

In [51]:
```
weather=df.groupby(['event' , pd.cut(df.windspeed,10)])
result= weather.size().unstack()
print(result)
```

```
windspeed  (4.985, 6.5]  (6.5, 8.0]  (8.0, 9.5]  (9.5, 11.0]  (11.0, 12.5]  \
event
Cloudy                0           1           0            1             0
Fog                   0           0           0            0             1
Rain                  2           0           0            0             0
Snow                  0           0           0            0             1
Sunny                 1           2           0            0             0

windspeed  (12.5, 14.0]  (14.0, 15.5]  (15.5, 17.0]  (17.0, 18.5]  \
event
Cloudy                1             0             0             0
Fog                   0             1             0             0
Rain                  0             0             0             0
Snow                  0             0             0             0
Sunny                 0             0             0             0

windspeed  (18.5, 20.0]
event
Cloudy                0
Fog                   0
Rain                  0
Snow                  0
Sunny                 1
```

In [ ]:
```
```

Question 7

In [52]:
```
df=pd.read_csv("Data_Q7.csv")
df
```

Out[52]:

	Name	Birth_Month	Gender	Pass_Division
0	Mudtt Chauhan	December	M	III
1	Seema Chopra	January	F	II
2	Rani Gupta	March	F	I
3	Aditya Narayan	October	M	I
4	Sanjeev Sahni	February	M	II
5	Prakash Kumar	December	M	III
6	Ritu Agarwal	September	F	I
7	Akshay Goel	August	M	I
8	Meeta Kulkarni	July	F	II
9	Preeti Ahuja	November	F	II
10	Sunit Oas Gupta	April	M	III
11	SonaliSapre	January	F	I
12	RashmiTalwar	June	F	III
13	Ashish Dubey	May	M	II
14	Kiran Sharma	February	F	II
15	Sameer Bansal	October	M	I

In [53]:
```
print(df['Gender'].unique())
print(df['Pass_Division'].unique())
```

```
['M' 'F']
['III' 'II' 'I']
```

part A

In [54]:
```
hot_encode=pd.get_dummies(df,columns=['Gender','Pass_Division'])
print(hot_encode)
```

	Name	Birth_Month	Gender_F	Gender_M	Pass_Division_I	Pass_Division_II	Pass_Division_III
0	Mudtt Chauhan	December	0	1	0	0	1
1	Seema Chopra	January	1	0	0	1	0
2	Rani Gupta	March	1	0	1	0	0
3	Aditya Narayan	October	0	1	1	0	0
4	Sanjeev Sahni	February	0	1	0	1	0
5	Prakash Kumar	December	0	1	0	0	1
6	Ritu Agarwal	September	1	0	1	0	0
7	Akshay Goel	August	0	1	1	0	0
8	Meeta Kulkarni	July	1	0	0	1	0
9	Preeti Ahuja	November	1	0	0	1	0
10	Sunit Oas Gupta	April	0	1	0	0	1
11	SonaliSapre	January	1	0	1	0	0
12	RashmiTalwar	June	1	0	0	0	1
13	Ashish Dubey	May	0	1	0	1	0
14	Kiran Sharma	February	1	0	0	1	0
15	Sameer Bansal	October	0	1	1	0	0

part B

In [55]:
```
sort=['January','February','March','April','May','June','July','August','September','October','November','December']
df['Birth_Month']=pd.Categorical(df['Birth_Month'],categories=sort,ordered=True)
df.sort_values('Birth_Month')
```

Out[55]:

	Name	Birth_Month	Gender	Pass_Division
1	Seema Chopra	January	F	II
11	SonaliSapre	January	F	I
4	Sanjeev Sahni	February	M	II
14	Kiran Sharma	February	F	II
2	Rani Gupta	March	F	I
10	Sunit Oas Gupta	April	M	III
13	Ashish Dubey	May	M	II
12	RashmiTalwar	June	F	III
8	Meeta Kulkarni	July	F	II
7	Akshay Goel	August	M	I
6	Ritu Agarwal	September	F	I
3	Aditya Narayan	October	M	I
15	Sameer Bansal	October	M	I
9	Preeti Ahuja	November	F	II
0	Mudtt Chauhan	December	M	III
5	Prakash Kumar	December	M	III

In [ ]:
```

```

Question 8

In [56]:
```
family =pd.read_csv('data10.csv')
family
```

Out[56]:

	Name	Gender	MonthlyIncome
0	Shah	Male	114000.0
1	Vats	Male	65000.0
2	Vats	Female	43150.0
3	Kumar	Female	69500.0
4	Vats	Female	155000.0
5	Kumar	Male	103000.0
6	Shah	Male	55000.0
7	Shah	Female	112400.0
8	Kumar	Female	81030.0
9	Vats	Male	71900.0

In [57]:
```
family.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Name           10 non-null     object 
 1   Gender         10 non-null     object 
 2   MonthlyIncome  10 non-null     float64
dtypes: float64(1), object(2)
memory usage: 372.0+ bytes
```

part A

In [58]:
```
familywise = family.groupby('Name')
familywise.describe()
```

Out[58]:

	MonthlyIncome							
	count	mean	std	min	25%	50%	75%	max
Name								
Kumar	3.0	84510.0	17018.968829	69500.0	75265.0	81030.0	92015.0	103000.0
Shah	3.0	93800.0	33611.307621	55000.0	83700.0	112400.0	113200.0	114000.0
Vats	4.0	83762.5	49047.279486	43150.0	59537.5	68450.0	92675.0	155000.0

In [59]:
```
familywise.sum("MonthlyIncome")
```

Out[59]:

	MonthlyIncome
Name	
Kumar	253530.0
Shah	281400.0
Vats	335050.0

part B

In [60]:
```
familywise.max()
```

Out[60]:

Name	Gender	MonthlyIncome
Kumar	Male	103000.0
Shah	Male	114000.0
Vats	Male	155000.0

part C

In [61]:
```
family[family['MonthlyIncome']>60000]
```

Out[61]:

	Name	Gender	MonthlyIncome
0	Shah	Male	114000.0
1	Vats	Male	65000.0
3	Kumar	Female	69500.0
4	Vats	Female	155000.0
5	Kumar	Male	103000.0
7	Shah	Female	112400.0
8	Kumar	Female	81030.0
9	Vats	Male	71900.0

part D

In [62]:
```
family[family['Gender']=='Female'].groupby('Name').mean("MonthlyIncome").loc['Shah']
```

Out[62]:
```
MonthlyIncome    112400.0
Name: Shah, dtype: float64
```

In [ ]:
```

```