GARIMA ROHILLA
CSC/21/37

QUESTION 1 :

Given below is a dictionary having two keys ‘Boys’ and ‘Girls’ and having two lists of
heights of five Boys and
Five Girls respectively as values associated with these keys
Original dictionary of lists:
{'Boys': [72, 68, 70, 69, 74], 'Girls': [63, 65, 69, 62, 61]}
From the given dictionary of lists create the following list of dictionaries:
[{'Boys': 72, 'Girls': 63}, {'Boys': 68, 'Girls': 65}, {'Boys': 70, 'Girls': 69}, {'Boys': 69, 'Girls':62}, {‘Boys’:74,‘Girls’:61]

```
a={'Boys':[72,68,70,69,74],'Girls':[63,65,69,62,61]}
rs=[]
for B,G in zip (a["Boys"],a["Girls"]):
    rs.append({"Boys":B,"Girls":G})
print(rs)
```

```
[{'Boys': 72, 'Girls': 63}, {'Boys': 68, 'Girls': 65}, {'Boys': 70, 'Girls': 69}, {'Boys': 6
9, 'Girls': 62}, {'Boys': 74, 'Girls': 61}]
```

GARIMA ROHILLA
CSC/21/37

PRACTICAL 2 :

Write programs in Python using NumPy library to do the following:

```
import numpy as np
```

a) Compute the mean, standard deviation, and variance of a two dimensional random integer array along the second axis.

```
import numpy as np

arr=np.random.randint(21,size=(4,3))
print("Random array : ")
print(arr)
arr_mean=arr.mean(axis=1)
print("Mean along second axis : ",arr_mean)
arr_std_dev=arr.std(axis=1)
print("Standard deviation along second axis : ",arr_std_dev)
arr_var=arr.var(axis=1)
print("Variance along second axis : ",arr_var)
```

```
Random array :
[[ 6  5 18]
 [ 9  5 13]
 [ 3  9 11]
 [16 14 15]]
Mean along second axis :  [ 9.66666667  9.          7.66666667 15.        ]
Standard deviation along second axis :  [5.90668172 3.26598632 3.39934634 0.81649658]
Variance along second axis :  [34.88888889 10.66666667 11.55555556  0.66666667]
```

b) Get the indices of the sorted elements of a given array.
   a. B = [56, 48, 22, 41, 78, 91, 24, 46, 8, 33]

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

```
array([8, 2, 6, 9, 3, 7, 1, 0, 4, 5], dtype=int32)
```

c) Create a 2-dimensional array of size m x n integer elements, also print the shape, type and data type of the array and then reshape it into nx m array, n and m are user inputs given at the run time.

```
import numpy as np

row=int(input("Enter no of rows(m) : "))
col=int(input("Enter no of col(n) : "))

array=np.random.randint(100,size=(row,col))
print("__Random 2d array__\n\n",array,"\n\n")

print("Shape of the array : ",np.shape(array))

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

print("Data type of the array : ",array.dtype)

print("__Reshaped array (n)X(m)__ : ",col,"X",row)
print(array.reshape(col,row))
```

```
= RESTART: C:/Users/HP/py/py.py
Enter no of rows(m) : 4
Enter no of col(n) : 3
__Random 2d array__

 [[57 66 68]
 [79 38 37]
 [32 24 69]
 [38 96 15]]


Shape of the array :  (4, 3)
Type of the array :  <class 'numpy.ndarray'>
Data type of the array :  int32
__Reshaped array (n)X(m)__ :  3 X 4
[[57 66 68 79]
 [38 37 32 24]
 [69 38 96 15]]
```

d) Test whether the elements of a given array are zero, non-zero and NaN. Record the indices of these elements in three separate arrays.

```
import numpy as np

ar=np.array([[4,8,0],[np.nan,3,4],[np.nan,4,2],[3,np.nan,0]])
print(ar)

r=np.argwhere(np.isnan(ar))
print("\n")
print(r)
print("\n")

r=np.argwhere(ar==0)
print(r)
print("\n")

r=np.argwhere(ar)
print(r)
print("\n")
```

GARIMA ROHILLA
CSC/21/37

```
[[ 4.  8.  0.]
 [nan  3.  4.]
 [nan  4.  2.]
 [ 3. nan  0.]]


[[1 0]
 [2 0]
 [3 1]]


[[0 2]
 [3 2]]


[[0 0]
 [0 1]
 [1 0]
 [1 1]
 [1 2]
 [2 0]
 [2 1]
 [2 2]
 [3 0]
 [3 1]]
```

NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37

PRACTICAL - 03

Create a dataframe having at least 3 columns and 50 rows to store numeric data generated using a random function. Replace 10% of the values by null values whose index positions are generated using random function.

a. Identify and count missing values in a dataframe.

```
import numpy as np
import pandas as pd
df=pd.DataFrame(np.random.randint(0,100,size=(50,3)),columns=['col1','col2','col3'])
df=df.mask(np.random.random(df.shape)<.1)
missing_values=df.isnull().sum()
print("Missing value: ",missing_values)
print("Total missing value: ",missing_values.sum())
```

[18] ✓ 0.0s

```
Missing value:  col1    6
col2    5
col3    9
dtype: int64
Total missing value:  20
```

b. Drop the column having more than 5 null values.

```
df=df.dropna(thresh=len(df)-5,axis=1)
print(df)
```

[20] ✓ 0.0s

NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37

```
    col2
0   22.0
1   29.0
2    NaN
3   12.0
4    NaN
5   60.0
6    8.0
7   77.0
8   82.0
9   98.0
10   NaN
11   NaN
12  56.0
13  41.0
14  13.0
15  34.0
16  74.0
17   8.0
18  16.0
19  12.0
20   8.0
21  65.0
22  36.0
23   7.0
24  16.0
25  44.0
26  69.0
27  60.0
28  90.0
29  18.0
30  44.0
31  57.0
32  71.0
33   0.0
34  91.0
35  24.0
36   NaN
37  43.0
38  44.0
39  20.0
40  74.0
41  18.0
42  17.0
43  13.0
44   0.0
45  15.0
46  74.0
47  95.0
48  75.0
49   7.0
```

c. Identify the row label having maximum of the sum of all values in a row and drop that row.

```
mx_row_label=df.sum(axis=1).idxmax()
print("Dropped row no: ",mx_row_label,"having sum: ",df.sum(axis=1).max())
df=df.drop(mx_row_label)
```

[22] ✓ 0.0s

```
Dropped row no:  47 having sum:  95.0
```

d. Sort the dataframe on the basis of the first column.

NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37

```
[23]  df=df.sort_values(by=df.columns[0])
      print("After sorting: ")
      df.head()
```

✓ 0.0s

```
After sorting:
```

	col2
33	0.0
44	0.0
49	7.0
23	7.0
6	8.0

e. Remove all duplicates from the first column.

```
[24]  df=df.drop_duplicates(subset=df.columns[0])
      df.head()
```

✓ 0.0s

	col2
33	0.0
49	7.0
6	8.0
19	12.0
43	13.0

f. Detect the outliers and remove the rows having outliers.

NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37

```
a1=df.quantile(0.25)
a2=df.quantile(0.75)
i=a2-a1
df=df[-((df<(a1-1.5*i))|(df>(a2+1.5*i))).any(axis=1)]
df.head()
```

[10] ✓ 0.0s

	col3
9	2.0
18	6.0
34	7.0
4	16.0
0	18.0

g. Discretize second column and create 5 bins.

```
b=[0,20,40,60,80,100]
df[df.columns[0]]=pd.cut(df[df.columns[0]],bins=b)
df[df.columns[0]]
```

[13] ✓ 0.0s

```
9       (0.0, 20.0]
18      (0.0, 20.0]
34      (0.0, 20.0]
4       (0.0, 20.0]
0       (0.0, 20.0]
40      (0.0, 20.0]
37     (20.0, 40.0]
24     (20.0, 40.0]
39     (20.0, 40.0]
41     (20.0, 40.0]
13     (20.0, 40.0]
30     (20.0, 40.0]
10     (20.0, 40.0]
21     (20.0, 40.0]
31     (20.0, 40.0]
45     (40.0, 60.0]
20     (40.0, 60.0]
16     (40.0, 60.0]
38     (40.0, 60.0]
19     (40.0, 60.0]
44     (60.0, 80.0]
32     (60.0, 80.0]
42     (60.0, 80.0]
8      (60.0, 80.0]
36     (60.0, 80.0]
23     (60.0, 80.0]
14    (80.0, 100.0]
6     (80.0, 100.0]
15    (80.0, 100.0]
48    (80.0, 100.0]
17    (80.0, 100.0]
49    (80.0, 100.0]
12    (80.0, 100.0]
7     (80.0, 100.0]
27    (80.0, 100.0]
3               NaN
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

PRACTICAL - 04

4. Consider two excel files having attendance of a workshop's participants for two days. Each file has three fields 'Name', 'Time of joining', duration (in minutes) where names are unique within a file. Note that duration may take one of three values (30, 40, 50) only. Import the data into two dataframes .

CODE :

```
[1]: import numpy as np
     import pandas as pd
```

```
[3]: df=pd.read_csv("Prac_4_Sheet.csv")
```

```
[4]: df.T
```

[4]:

	0	1	2	3	4	5	6	7

[37]:

	Name	Time of Joining	Duration	Day
0	Braelyn McIntosh	10:30	50	1
1	Kristian Sanders	11:30	40	1
2	Emma Strong	12:30	30	1
3	Axl Jaramillo	13:30	50	1
4	Guadalupe Rush	14:30	40	1
5	Kaiser Higgins	15:30	30	1
6	Leighton Tang	16:30	50	1
7	Rogelio Marin	17:30	40	1
8	Celia Pena	9:00	30	1
9	Marcus McDonald	10:00	50	1
10	Daisy Villegas	11:00	40	1
11	Kieran Golden	12:00	50	1
12	Giuliana Reynolds	13:00	50	1
13	Vincent Chung	14:00	50	1
14	Rivka McBride	15:00	50	1
15	Denver Compton	16:00	40	1
16	Elina Hawkins	11:00	30	1
17	Victor Santana	12:00	50	1
18	Myra Holt	13:00	40	1
19	Niko Pena	14:00	30	1
20	Rachel Moran	8:30	30	1
21	Tate Harper	9:30	50	1
22	Ana Ward	10:30	50	1
23	Jameson Chambers	11:30	40	1
24	Makayla Bauer	12:30	50	1
25	Kieran Blackwell	13:30	50	1
26	Saoirse McCall	14:30	50	1
27	Kiaan Gilbert	15:30	40	1
28	Jocelyn Sawyer	16:30	30	2
29	Jefferson Wilson	17:30	50	2
30	Luna McCann	10:00	40	2
31	Heath Mathis	11:00	30	2
32	Anne Wells	12:00	50	2
33	Max Snyder	13:00	40	2
34	Callie Moses	14:00	30	2
35	Niklaus Frost	15:00	50	2
36	Paula Park	16:00	40	2
37	Enzo Farley	17:00	30	2
38	Wrenley Cochran	10:15	50	2

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
[5]: df1=df[df['Day']==1].drop('Day',axis=1)
     df2=df[df['Day']==2].drop('Day',axis=1)
```

Day 1 Participants

```
[6]: df1
```

[6]:

	Name	Time of Joining	Duration
0	Braelyn McIntosh	10:30	50
1	Kristian Sanders	11:30	40
2	Emma Strong	12:30	30
3	Axl Jaramillo	13:30	50
4	Guadalupe Rush	14:30	40
5	Kaiser Higgins	15:30	30
6	Leighton Tang	16:30	50
7	Rogelio Marin	17:30	40
8	Celia Pena	9:00	30
9	Marcus McDonald	10:00	50
10	Daisy Villegas	11:00	40
11	Kieran Golden	12:00	50
12	Giuliana Reynolds	13:00	50
13	Vincent Chung	14:00	50
14	Rivka McBride	15:00	50
15	Denver Compton	16:00	40
16	Elina Hawkins	11:00	30
17	Victor Santana	12:00	50
18	Myra Holt	13:00	40
19	Niko Pena	14:00	30
20	Rachel Moran	8:30	30
21	Tate Harper	9:30	50
22	Ana Ward	10:30	50
23	Jameson Chambers	11:30	40
24	Makayla Bauer	12:30	50
25	Kieran Blackwell	13:30	50
26	Saoirse McCall	14:30	50

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

Day 2 Participants

```
[7]: df2
```

[7]:

	Name	Time of Joining	Duration
28	Jocelyn Sawyer	16:30	30
29	Jefferson Wilson	17:30	50
30	Luna McCann	10:00	40
31	Heath Mathis	11:00	30
32	Anne Wells	12:00	50
33	Max Snyder	13:00	40
34	Callie Moses	14:00	30
35	Niklaus Frost	15:00	50
36	Paula Park	16:00	40
37	Enzo Farley	17:00	30
38	Wrenley Cochran	10:15	50
39	Danny Johnson	11:15	40
40	Emma Strong	12:15	30
41	Axl Jaramillo	13:15	30
42	Guadalupe Rush	14:15	50
43	Kaiser Higgins	15:15	40
44	Leighton Tang	16:15	30
45	Rogelio Marin	17:15	40
46	Celia Pena	13:00	30
47	Marcus McDonald	14:00	50
48	Daisy Villegas	15:00	40

a) Perform merging of the two dataframes to find the names of students who had attended the workshop on both days.

CODE :

```
[8]: pd.merge(df1,df2,'inner',"Name")
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

[8]:

	Name	Time of Joining_x	Duration_x	Time of Joining_y	Duration_y
0	Emma Strong	12:30	30	12:15	30
1	Axl Jaramillo	13:30	50	13:15	30
2	Guadalupe Rush	14:30	40	14:15	50
3	Kaiser Higgins	15:30	30	15:15	40
4	Leighton Tang	16:30	50	16:15	30
5	Rogelio Marin	17:30	40	17:15	40
6	Celia Pena	9:00	30	13:00	30
7	Marcus McDonald	10:00	50	14:00	50
8	Daisy Villegas	11:00	40	15:00	40

b) Find names of all students who have attended workshop on either of the days.

CODE :

```
[9]: pd.merge(df1,df2,'outer').Name
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
[9]:  0      Braelyn McIntosh
      1     Kristian Sanders
      2          Emma Strong
      3        Axl Jaramillo
      4      Guadalupe Rush
      5      Kaiser Higgins
      6       Leighton Tang
      7       Rogelio Marin
      8          Celia Pena
      9      Marcus McDonald
      10      Daisy Villegas
      11       Kieran Golden
      12    Giuliana Reynolds
      13       Vincent Chung
      14       Rivka McBride
      15      Denver Compton
      16       Elina Hawkins
      17      Victor Santana
      18           Myra Holt
      19           Niko Pena
      20       Rachel Moran
      21        Tate Harper
      22           Ana Ward
      23    Jameson Chambers
      24       Makayla Bauer
      25    Kieran Blackwell
      26      Saoirse McCall
      27       Kiaan Gilbert
      28      Jocelyn Sawyer
      29    Jefferson Wilson
      30          Luna McCann
      31       Heath Mathis
      32          Anne Wells
      33          Max Snyder
      34       Callie Moses
      35       Niklaus Frost
      36          Paula Park
      37        Enzo Farley
      38     Wrenley Cochran
      39      Danny Johnson
      40         Emma Strong
      41       Axl Jaramillo
      42     Guadalupe Rush
      43     Kaiser Higgins
      44      Leighton Tang
      45      Rogelio Marin
      46         Celia Pena
      47     Marcus McDonald
```

c) Merge two data frames row-wise and find the total number of records in the data frame.

CODE :

```
[10]: len(pd.concat([df2,df1],axis=0,ignore_index=1))

[10]: 49
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

d) Merge two data frames and use two columns names and duration as multi-row indexes. Generate descriptive statistics for this multi-index.

CODE :

```
[11]: pd.concat([df1,df2],axis=0)
```

[11]:

	Name	Time of Joining	Duration
0	Braelyn McIntosh	10:30	50
1	Kristian Sanders	11:30	40
2	Emma Strong	12:30	30
3	Axl Jaramillo	13:30	50
4	Guadalupe Rush	14:30	40
5	Kaiser Higgins	15:30	30
6	Leighton Tang	16:30	50
7	Rogelio Marin	17:30	40
8	Celia Pena	9:00	30
9	Marcus McDonald	10:00	50
10	Daisy Villegas	11:00	40
11	Kieran Golden	12:00	50
12	Giuliana Reynolds	13:00	50
13	Vincent Chung	14:00	50
14	Rivka McBride	15:00	50
15	Denver Compton	16:00	40
16	Elina Hawkins	11:00	30
17	Victor Santana	12:00	50
18	Myra Hoit	13:00	40
19	Niko Pena	14:00	30
20	Rachel Moran	8:30	30
21	Tate Harper	9:30	50
22	Ana Ward	10:30	50
23	Jameson Chambers	11:30	40
24	Makayla Bauer	12:30	50
25	Kieran Blackwell	13:30	50
26	Saoirse McCall	14:30	50
27	Kiaan Gilbert	15:30	40
28	Jocelyn Sawyer	16:30	30

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
[12]: Df3=pd.concat([df1,df2],axis=0).set_index(['Name','Duration'])
```

```
[13]: Df3
```

[13]:

		Time of Joining
Name	Duration	
Braelyn McIntosh	50	10:30
Kristian Sanders	40	11:30
Emma Strong	30	12:30
Axl Jaramillo	50	13:30
Guadalupe Rush	40	14:30
Kaiser Higgins	30	15:30
Leighton Tang	50	16:30
Rogelio Marin	40	17:30
Celia Pena	30	9:00
Marcus McDonald	50	10:00
Daisy Villegas	40	11:00
Kieran Golden	50	12:00
Giuliana Reynolds	50	13:00
Vincent Chung	50	14:00
Rivka McBride	50	15:00
Denver Compton	40	16:00
Elina Hawkins	30	11:00
Victor Santana	50	12:00
Myra Holt	40	13:00
Niko Pena	30	14:00
Rachel Moran	30	8:30
Tate Harper	50	9:30
Ana Ward	50	10:30
Jameson Chambers	40	11:30
Makayla Bauer	50	12:30
Kieran Blackwell	50	13:30
Saoirse McCall	50	14:30

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
[16]: Df3.describe()
```

[16]:

	Time of Joining
count	49
unique	27
top	14:00
freq	4

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

PRACTICAL - 05

5. Taking Iris data, plot the following with proper legend and axis labels: (Download IRIS data from: https://archive.ics.uci.edu/ml/datasets/iris (https://archive.ics.uci.edu/ml/datasets/iris) or import it from sklearn.datasets)

a. Plot bar chart to show the frequency of each class label in the data.

b. Draw a scatter plot for Petal width vs sepal width.

c. Plot density distribution for feature petal length.

d. Use a pair plot to show pairwise bivariate distribution in the Iris Dataset.

CODE :

[90]
```
import numpy as np
import pandas as pd
import sklearn.datasets as sns
import matplotlib.pyplot as plt
import seaborn as sns
```

[73]
```
iris=sns.load_iris()
```

[74]
```
print(iris.DESCR)
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes and the class
    :Attribute Information:
        - sepal length in cm
        - sepal width in cm
        - petal length in cm
        - petal width in cm
        - class:
                - Iris-Setosa
                - Iris-Versicolour
                - Iris-Virginica

    :Summary Statistics:

    ============== ==== ==== ======= ===== ====================
                    Min  Max   Mean    SD   Class Correlation
    ============== ==== ==== ======= ===== ====================
    sepal length:   4.3  7.9   5.84   0.83    0.7826
    sepal width:    2.0  4.4   3.05   0.43   -0.4194
    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
    ============== ==== ==== ======= ===== ====================
    :Missing Attribute Values: None
    :Class Distribution: 33.3% for each of 3 classes.
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988

The famous Iris database, first used by Sir R.A. Fisher. The dataset is taken
from Fisher's paper. Note that it's the same as in R, but not as in the UCI
Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the
pattern recognition literature.  Fisher's paper is a classic in the field and
is referenced frequently to this day.  (See Duda & Hart, for example.)  The
data set contains 3 classes of 50 instances each, where each class refers to a
type of iris plant.  One class is linearly separable from the other 2; the
latter are NOT linearly separable from each other.

.. topic:: References

   - Fisher, R.A. "The use of multiple measurements in taxonomic problems"
     Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to
     Mathematical Statistics" (John Wiley, NY, 1950).
   - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis.
     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.
   - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System
     Structure and Classification Rule for Recognition in Partially Exposed
     Environments".  IEEE Transactions on Pattern Analysis and Machine
     Intelligence, Vol. PAMI-2, No. 1, 67-71.
   - Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule".  IEEE Transactions
     on Information Theory, May 1972, 431-433.
   - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLASS II
     conceptual clustering system finds 3 classes in the data.
   - Many, many more ...
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

[75]
```
iris.feature_names
```

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

[76]
```
iris.target
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

[77]
```
df=pd.DataFrame(data=iris.data,columns=iris.feature_names)
```

[78]
```
df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

[79]
```
df['Iris type'] = iris['target']
```

[80]
```
df
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Iris type
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

[81]
```
def convert(x):
    if x == 0:
        val = 'setosa'
    elif x == 1:
        val = 'versicolor'
    else:
        val = 'virginica'
    return val
```

[86]
```
df['type name']=df['Iris type'].apply(convert)
```

[87]
```
df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Iris type	type name
0	5.1	3.5	1.4	0.2	0	setosa
1	4.9	3.0	1.4	0.2	0	setosa
2	4.7	3.2	1.3	0.2	0	setosa
3	4.6	3.1	1.5	0.2	0	setosa
4	5.0	3.6	1.4	0.2	0	setosa

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

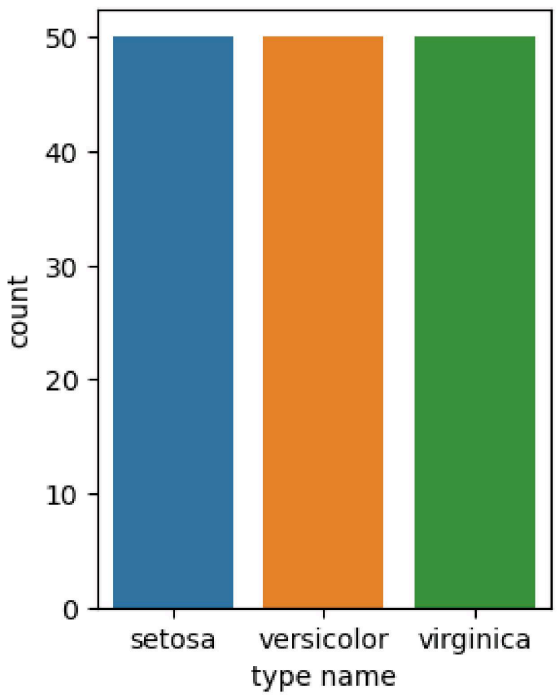
a. Plot bar chart to show the frequency of each class label in the data.

[89]
```
df.groupby('type name').count()
```

type name	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Iris type
setosa	50	50	50	50	50
versicolor	50	50	50	50	50
virginica	50	50	50	50	50

[97]
```
plt.figure(figsize=(3,4))
sns.countplot(x='type name',data=df)
```

```
<Axes: xlabel='type name', ylabel='count'>
```



SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

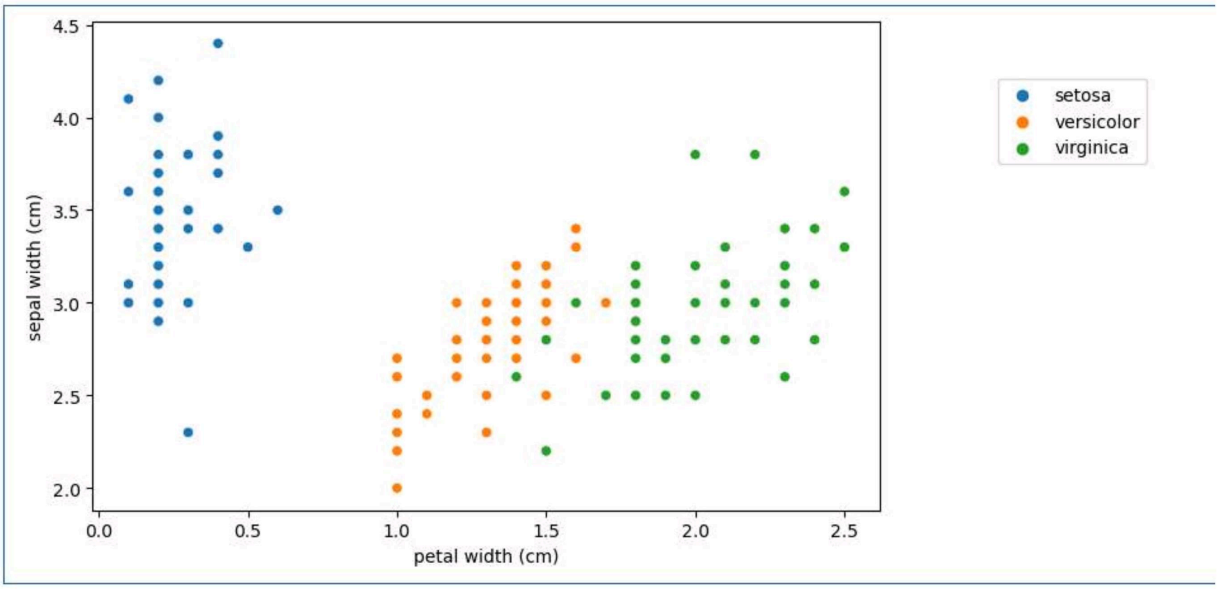
b. Draw a scatter plot for Petal width vs sepal width.

[98]
```
df.columns
```

```
Index(['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)',
       'petal width (cm)', 'Iris type', 'type name'],
      dtype='object')
```

[108]
```
plt.figure(figsize=(8,5))
sns.scatterplot(x='petal width (cm)',y='sepal width (cm)',data=df,hue='type name')
plt.legend(bbox_to_anchor=(1.35,0.9))
```

```
<matplotlib.legend.Legend at 0x12c41ab90>
```

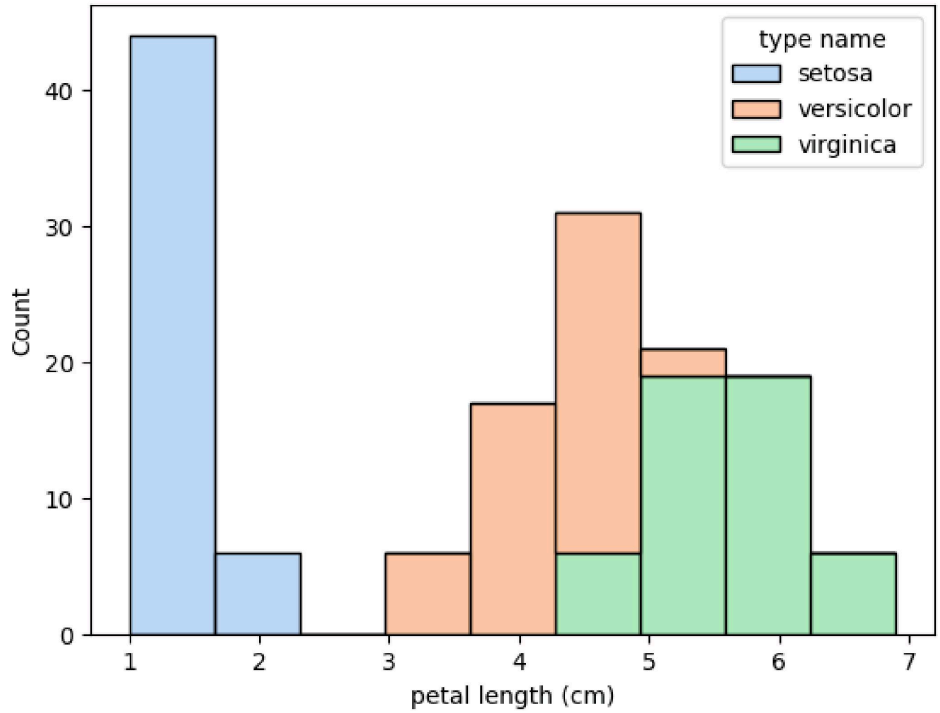


SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

c. Plot density distribution for feature petal length.

```
[127]  sns.histplot(x='petal length (cm)',data=df,hue='type name',multiple='stack',palette='pastel')
```

```
<Axes: xlabel='petal length (cm)', ylabel='Count'>
```

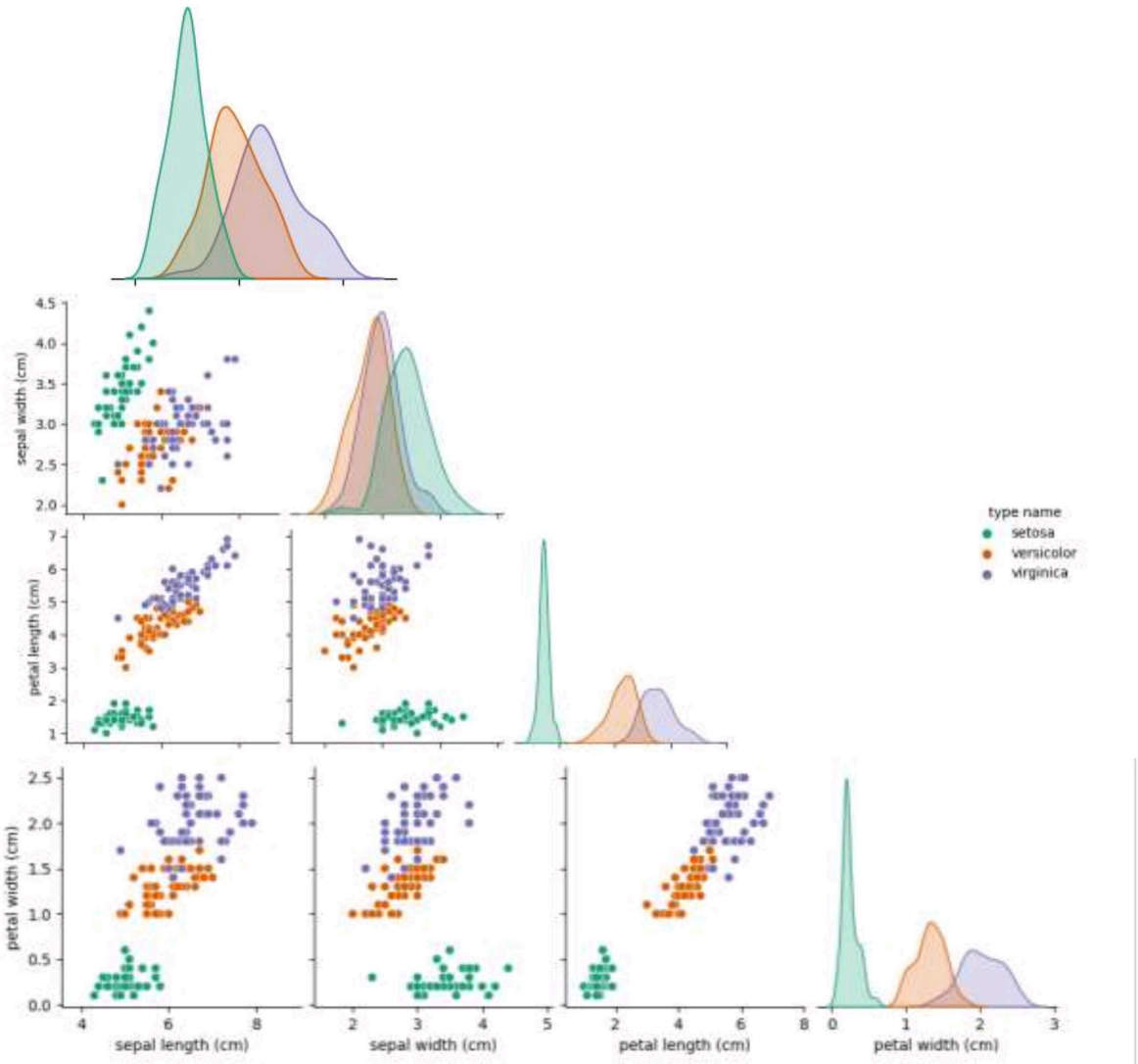


SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

d. Use a pair plot to show pairwise bivariate distribution in the Iris Dataset.

```
[133]  sns.pairplot(df.drop('Iris type',axis=1),corner=1,hue='type name',palette='Dark2')
```

```
...  <seaborn.axisgrid.PairGrid at 0x16af3b280>
```



SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

PRACTICAL - 06

6. Consider any sales training/ weather forecasting dataset :

CODE :

In [1]:
```
# importing numpy  library
# importing pandas library
import numpy as np
import pandas as pd
```

In [2]:
```
# Loading dataset into a DataFrame
data = pd.read_excel("Sales.xlsx")
data.head(10)
```

Out[2]:

	Order ID	Order Date	Ship yearMonth	State	Postal Code	Region	Product ID	Category	Sub-Category	
0	CA-2017-152156	2017-11-08	2017-11	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	26
1	CA-2017-152156	2017-11-08	2017-11	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	73
2	CA-2017-138688	2017-06-12	2017-06	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	1
3	US-2016-108966	NaT	2016-10	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	95
4	US-2016-108966	2016-10-11	2016-10	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	2
5	CA-2015-115812	2015-06-09	2015-06	California	90032	West	FUR-FU-10001487	Furniture	Furnishings	4
6	CA-2015-115812	2015-06-09	2015-06	California	90032	West	OFF-AR-10002833	Office Supplies	Art	
7	CA-2015-115812	2015-06-09	2015-06	California	90032	West	TEC-PH-10002275	Technology	Phones	90
8	CA-2015-115812	2015-06-09	2015-06	California	90032	West	OFF-BI-10003910	Office Supplies	Binders	1
9	CA-2015-115812	NaT	2015-06	California	90032	West	OFF-AP-10002892	Office Supplies	Appliances	11

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

a. Compute mean of a series grouped by another series.

```
In [3]: grouped_mean = data.groupby('State')['Sales'].mean()

        print("The Mean of a Series grouped by another :")
        grouped_mean
```

```
The Mean of a Series grouped by another :

Out[3]: State
        California        415.27600
        Florida           489.97275
        Kentucky          496.95000
        North Carolina     15.55200
        Texas              35.67700
        Utah               55.50000
        Washington        407.97600
        Wisconsin         665.88000
        Name: Sales, dtype: float64
```

b. Fill an intermittent time series to replace all missing dates with values of previous non-missing date.

```
In [4]: #changing into date format
        data['Order Date'] = pd.to_datetime(data['Order Date'])

        # Forward-fill missing values
        data['Order Date'] = data['Order Date'].ffill()
        data.head(10)
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

Out[4]:

	Order ID	Order Date	Ship yearMonth	State	Postal Code	Region	Product ID	Category	Sub-Category	
0	CA-2017-152156	2017-11-08	2017-11	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	26
1	CA-2017-152156	2017-11-08	2017-11	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	73
2	CA-2017-138688	2017-06-12	2017-06	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	1·
3	US-2016-108966	2017-06-12	2016-10	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	95
4	US-2016-108966	2016-10-11	2016-10	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	2
5	CA-2015-115812	2015-06-09	2015-06	California	90032	West	FUR-FU-10001487	Furniture	Furnishings	4
6	CA-2015-115812	2015-06-09	2015-06	California	90032	West	OFF-AR-10002833	Office Supplies	Art	
7	CA-2015-115812	2015-06-09	2015-06	California	90032	West	TEC-PH-10002275	Technology	Phones	90
8	CA-2015-115812	2015-06-09	2015-06	California	90032	West	OFF-BI-10003910	Office Supplies	Binders	1
9	CA-2015-115812	2015-06-09	2015-06	California	90032	West	OFF-AP-10002892	Office Supplies	Appliances	11·

c. Perform appropriate year-month string to dates conversion.

In [5]:
```
#conveting into date time

data['Ship yearMonth'] = pd.to_datetime(data['Ship yearMonth'])
data.head(10)
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

Out[5]:

	Order ID	Order Date	Ship yearMonth	State	Postal Code	Region	Product ID	Category	Sub-Category	
0	CA-2017-152156	2017-11-08	2017-11-01	Kentucky	42420	South	FUR-BO-10001798	Furniture	Bookcases	26
1	CA-2017-152156	2017-11-08	2017-11-01	Kentucky	42420	South	FUR-CH-10000454	Furniture	Chairs	73
2	CA-2017-138688	2017-06-12	2017-06-01	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	1·
3	US-2016-108966	2017-06-12	2016-10-01	Florida	33311	South	FUR-TA-10000577	Furniture	Tables	95
4	US-2016-108966	2016-10-11	2016-10-01	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	2
5	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	FUR-FU-10001487	Furniture	Furnishings	4
6	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	OFF-AR-10002833	Office Supplies	Art	
7	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	TEC-PH-10002275	Technology	Phones	90
8	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	OFF-BI-10003910	Office Supplies	Binders	1
9	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	OFF-AP-10002892	Office Supplies	Appliances	11·

d. Split a dataset to group by two columns and then sort the aggregated results within the groups.

In [6]:
```
#Lambda function for sorting
func = lambda x: x.sort_values(by='Sales')

sorted_data = data.groupby(['State', 'Region']).apply(func).reset_index(dro
p=True)
sorted_data.head(10)
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

Out[6]:

	Order ID	Order Date	Ship yearMonth	State	Postal Code	Region	Product ID	Category	Sub-Category	
0	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	OFF-AR-10002833	Office Supplies	Art	
1	CA-2015-143336	2015-08-27	2015-09-01	California	94109	West	OFF-AR-10003056	Office Supplies	Art	
2	CA-2017-138688	2017-06-12	2017-06-01	California	90036	West	OFF-LA-10000240	Office Supplies	Labels	
3	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	OFF-BI-10003910	Office Supplies	Binders	
4	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	FUR-FU-10001487	Furniture	Furnishings	
5	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	OFF-AP-10002892	Office Supplies	Appliances	1
6	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	TEC-PH-10002275	Technology	Phones	9
7	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	TEC-PH-10002033	Technology	Phones	9
8	CA-2015-115812	2015-06-09	2015-06-01	California	90032	West	FUR-TA-10001539	Furniture	Tables	17
9	US-2016-108966	2016-10-11	2016-10-01	Florida	33311	South	OFF-ST-10000760	Office Supplies	Storage	

e. Split a given dataframe into groups with bin counts.

In [9]:
```
#creating  bins
bin = pd.cut(data['Sales'], bins=3)

grouped_data = data.groupby(bin, observed=False)
for key, group in grouped_data:
    print("Key : ",key)
    print("GROUP \n",group)
    print("\n\n-----------------------------------")
    print("-----------------------------------\n\n")
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
Key :  (0.84, 570.424]
GROUP
```

	Order ID	Order Date	Ship yearMonth	State	Postal Code
0	CA-2017-152156	2017-11-08	2017-11-01	Kentucky	42420
2	CA-2017-138688	2017-06-12	2017-06-01	California	90036
4	US-2016-108966	2016-10-11	2016-10-01	Florida	33311
5	CA-2015-115812	2015-06-09	2015-06-01	California	90032
6	CA-2015-115812	2015-06-09	2015-06-01	California	90032
8	CA-2015-115812	2015-06-09	2015-06-01	California	90032
9	CA-2015-115812	2015-06-09	2015-06-01	California	90032
12	CA-2018-114412	2018-04-15	2018-04-01	North Carolina	28027
13	CA-2017-161389	2018-04-15	2017-12-01	Washington	98103
14	US-2016-118983	2016-11-22	2016-11-01	Texas	76106
15	US-2016-118983	2016-11-22	2016-11-01	Texas	76106
17	CA-2015-167164	2015-05-13	2015-05-01	Utah	84084
18	CA-2015-143336	2015-08-27	2015-09-01	California	94109

	Region	Product ID	Category	Sub-Category	Sales
0	South	FUR-BO-10001798	Furniture	Bookcases	261.960
2	West	OFF-LA-10000240	Office Supplies	Labels	14.620
4	South	OFF-ST-10000760	Office Supplies	Storage	22.368
5	West	FUR-FU-10001487	Furniture	Furnishings	48.860
6	West	OFF-AR-10002833	Office Supplies	Art	7.280
8	West	OFF-BI-10003910	Office Supplies	Binders	18.504
9	West	OFF-AP-10002892	Office Supplies	Appliances	114.900
12	South	OFF-PA-10002365	Office Supplies	Paper	15.552
13	West	OFF-BI-10003656	Office Supplies	Binders	407.976
14	Central	OFF-AP-10002311	Office Supplies	Appliances	68.810
15	Central	OFF-BI-10000756	Office Supplies	Binders	2.544
17	West	OFF-ST-10000107	Office Supplies	Storage	55.500
18	West	OFF-AR-10003056	Office Supplies	Art	8.560

```
-------------------------------------
-------------------------------------

Key :  (570.424, 1138.304]
GROUP
```

	Order ID	Order Date	Ship yearMonth	State	Postal Code	\
1	CA-2017-152156	2017-11-08	2017-11-01	Kentucky	42420	
3	US-2016-108966	2017-06-12	2016-10-01	Florida	33311	
7	CA-2015-115812	2015-06-09	2015-06-01	California	90032	
11	CA-2015-115812	2015-06-09	2015-06-01	California	90032	
16	CA-2015-105893	2016-11-22	2015-11-01	Wisconsin	53711	

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

```
     Region       Product ID         Category Sub-Category     Sales
1     South  FUR-CH-10000454        Furniture       Chairs  731.9400
3     South  FUR-TA-10000577        Furniture       Tables  957.5775
7      West  TEC-PH-10002275       Technology       Phones  907.1520
11     West  TEC-PH-10002033       Technology       Phones  911.4240
16  Central  OFF-ST-10004186  Office Supplies      Storage  665.8800


------------------------------------
------------------------------------


Key :  (1138.304, 1706.184]
GROUP
           Order ID Order Date Ship yearMonth       State  Postal Code Regi
on  \
10  CA-2015-115812 2015-06-09     2015-06-01  California        90032   Wes
t

         Product ID   Category Sub-Category     Sales
10  FUR-TA-10001539  Furniture       Tables  1706.184


------------------------------------
------------------------------------
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

PRACTICAL - 07

7. Consider a data frame containing data about students i.e. name, gender and passing division:

	Name	Birth_Month	Gender	Pass_Division
0	Mudit 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	Sunil Das Gupta	April	M	III
11	Sonali Sapre	January	F	I
12	Rashmi Talwar	June	F	III
13	Ashish Dubey	May	M	II
14	Kiran Sharma	February	F	II
15	Sameer Bansal	October	M	I

a. Perform one hot encoding of the last two columns of categorical data using the get_dummies() function.

b. Sort this data frame on the “Birth Month” column (i.e. January to December). Hint: Convert Month to Categorical.

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

CODE :

```
import pandas as pd
```

[2] ✓ 0.0s

```
df=pd.DataFrame({
    'Name':['Mohit Chauhan','Seema Chopra','Rani Gupta','Aditya Narayan','Sanjeev Sahni',
            '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','September',
                   'August','July','November','April','January','June','May','February','October'],
    '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
```

[8] ✓ 0.0s

...

	Name	Birth_Month	Gender	Pass_Division
0	Mohit 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	Sunil Das Gupta	April	M	III
11	Sonali Sapre	January	F	I
12	Rashmi Talwar	June	F	III
13	Ashish Dubey	May	M	II
14	Kiran Sharma	February	F	II
15	Sameer Bansal	October	M	I

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

a. Perform one hot encoding of the last two columns of categorical data using the get_dummies() function.

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

[10] ✓ 0.0s

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

b. Sort this data frame on the “Birth Month” column (i.e. January to December). Hint: Convert Month to Categorical.

```
month_order=['January','February','March','April','May','June',
             'July','August','September','October','November','December']
df_encoded['Birth_Month']=pd.Categorical(df_encoded['Birth_Month'],
            categories=month_order,ordered=True)
df_sorted=df_encoded.sort_values(by='Birth_Month')
df_sorted
```

[11] ✓ 0.0s

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

...

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

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

PRACTICAL - 08

8. Consider the following data frame containing a family name, gender of the family member and her/his monthly income in each record. Name Gender MonthlyIncome (Rs.)

Name	Gender	MonthlyIncome (Rs.)
Shah	Male	114000.00
Vats	Male	65000.00
Vats	Female	43150.00
Kumar	Female	69500.00
Vats	Female	155000.00
Kumar	Male	103000.00
Shah	Male	55000.00
Shah	Female	112400.00
Kumar	Female	81030.00
Vats	Male	71900.00

CODE :

```
import numpy as np
import pandas as pd
```

[1] ✓ 3.3s

```
data={'Name':['Shah','Vats','Vats','Kumar','Vats','Kumar','Shah','Shah','Kumar','Vats'],
      'Gender':['Male','Male','Female','Female','Female','Male','Male','Female','Female','Male'],
      'MonthlyIncome (Rs.)':[114000.00,65000.00,43150.00,69500.00,155000.00,
                             103000.00,55000.00,112400.00,81030.00,71900.00]
      }
df=pd.DataFrame(data)
df
```

[2] ✓ 0.0s

	Name	Gender	MonthlyIncome (Rs.)
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

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

a. Calculate and display familywise gross monthly income.

```
[3]  familywise_income=df.groupby('Name')['MonthlyIncome (Rs.)'].sum()
     print("Familywise Gross Monthly Income:")
     print(familywise_income)
```

✓ 0.0s

```
Familywise Gross Monthly Income:
Name
Kumar    253530.0
Shah     281400.0
Vats     335050.0
Name: MonthlyIncome (Rs.), dtype: float64
```

b. Calculate and display the member with the highest monthly income in a family.

```
[5]  max_income=df.groupby('Name')['MonthlyIncome (Rs.)'].idxmax()
     highest_income=df.loc[max_income]
     print("Member with Highest Monthly Income in each Family:")
     print(highest_income)
```

✓ 0.0s

```
Member with Highest Monthly Income in each Family:
    Name  Gender  MonthlyIncome (Rs.)
5  Kumar    Male             103000.0
0   Shah    Male             114000.0
4   Vats  Female             155000.0
```

c. Calculate and display monthly income of all members with income greater than Rs. 60000.00.

```
[6]  high_income_members=df[df['MonthlyIncome (Rs.)']>60000.00]
     print("Monthly Income of members with Income > Rs. 60000.00:")
     print(high_income_members[['Name','MonthlyIncome (Rs.)']])
```

✓ 0.0s

```
Monthly Income of members with Income > Rs. 60000.00:
    Name  MonthlyIncome (Rs.)
0   Shah             114000.0
1   Vats              65000.0
3  Kumar              69500.0
4   Vats             155000.0
5  Kumar             103000.0
7   Shah             112400.0
8  Kumar              81030.0
9   Vats              71900.0
```

SUBMITTED TO : DR. PRITI JAGWANI
NAME : GARIMA ROHILLA
ROLL NO : CSC/21/37
UNIVERSITY ROLL N0 : 21059570010

d. Calculate and display the average monthly income of the female members in the Shah family.

```
[7]  shah_fam_avg_income=df[(df['Name']=='Shah')&(df['Gender']=='Female')]['MonthlyIncome (Rs.)'].mean()
     print("Average Monthly Income of Female Members in Shsh Family:",shah_fam_avg_income)
     ✓ 0.0s

...  Average Monthly Income of Female Members in Shsh Family: 112400.0
```