INTRODUCTION TO MACHINE LEARNING

LAB ASSIGNMENT – 1

BASIC ML EXERCISES

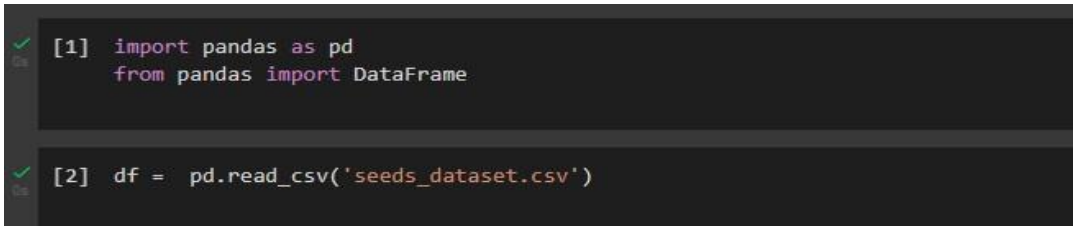
NAME : PRATHAPANI SATWIKA

REG.NO. : 20BCD7160

Write down the descriptions for the commands used in each questions

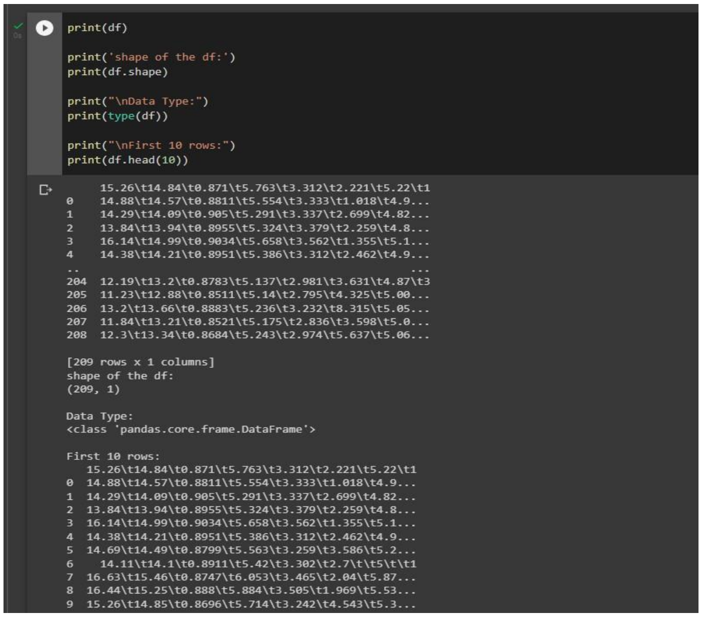
1. Write a Python program to load the wheat seeds data from a given csv file into a dataframe and print the shape of the data, type of the data and first 10 rows.

CODE AND OUTPUT:



```
[1]  import pandas as pd
     from pandas import DataFrame
```

```
[2]  df =  pd.read_csv('seeds_dataset.csv')
```

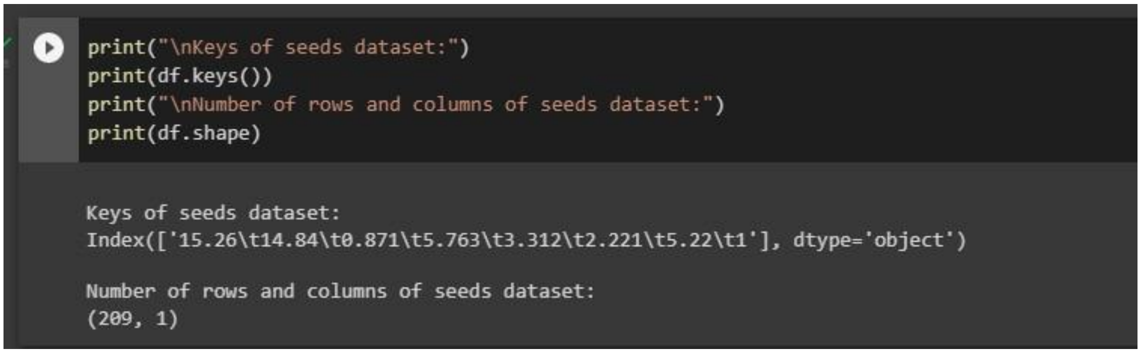


```
print(df)

print('shape of the df:')
print(df.shape)

print("\nData Type:")
print(type(df))

print("\nFirst 10 rows:")
print(df.head(10))
```

```
      15.26\t14.84\t0.871\t5.763\t3.312\t2.221\t5.22\t1
0     14.88\t14.57\t0.8811\t5.554\t3.333\t1.018\t4.9...
1     14.29\t14.09\t0.905\t5.291\t3.337\t2.699\t4.82...
2     13.84\t13.94\t0.8955\t5.324\t3.379\t2.259\t4.8...
3     16.14\t14.99\t0.9034\t5.658\t3.562\t1.355\t5.1...
4     14.38\t14.21\t0.8951\t5.386\t3.312\t2.462\t4.9...
..                                                 ...
204   12.19\t13.2\t0.8783\t5.137\t2.981\t3.631\t4.87\t3
205   11.23\t12.88\t0.8511\t5.14\t2.795\t4.325\t5.00...
206   13.2\t13.66\t0.8883\t5.236\t3.232\t8.315\t5.05...
207   11.84\t13.21\t0.8521\t5.175\t2.836\t3.598\t5.0...
208   12.3\t13.34\t0.8684\t5.243\t2.974\t5.637\t5.06...

[209 rows x 1 columns]
shape of the df:
(209, 1)

Data Type:
<class 'pandas.core.frame.DataFrame'>

First 10 rows:
   15.26\t14.84\t0.871\t5.763\t3.312\t2.221\t5.22\t1
0  14.88\t14.57\t0.8811\t5.554\t3.333\t1.018\t4.9...
1  14.29\t14.09\t0.905\t5.291\t3.337\t2.699\t4.82...
2  13.84\t13.94\t0.8955\t5.324\t3.379\t2.259\t4.8...
3  16.14\t14.99\t0.9034\t5.658\t3.562\t1.355\t5.1...
4  14.38\t14.21\t0.8951\t5.386\t3.312\t2.462\t4.9...
5  14.69\t14.49\t0.8799\t5.563\t3.259\t3.586\t5.2...
6    14.11\t14.1\t0.8911\t5.42\t3.302\t2.7\t\t5\t\t1
7  16.63\t15.46\t0.8747\t6.053\t3.465\t2.04\t5.87...
8  16.44\t15.25\t0.888\t5.884\t3.505\t1.969\t5.53...
9  15.26\t14.85\t0.8696\t5.714\t3.242\t4.543\t5.3...
```

2. Write a Python program using Scikit-learn to print the keys, number of rows-columns, feature names and the description of the dataset

CODE AND OUTPUT:

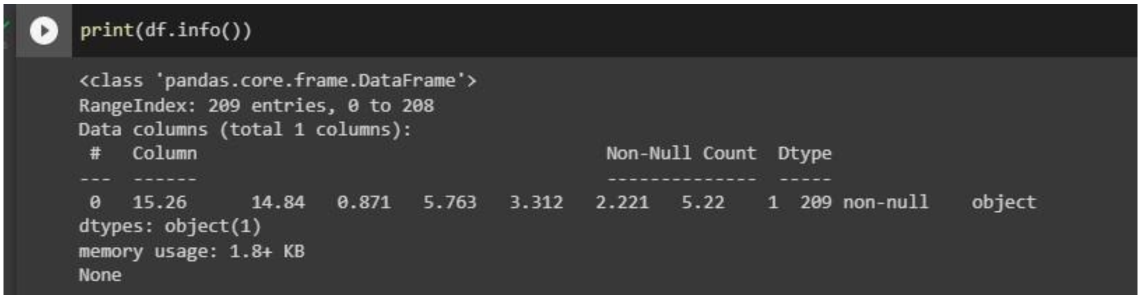


```
print("\nKeys of seeds dataset:")
print(df.keys())
print("\nNumber of rows and columns of seeds dataset:")
print(df.shape)
```

```
Keys of seeds dataset:
Index(['15.26\t14.84\t0.871\t5.763\t3.312\t2.221\t5.22\t1'], dtype='object')

Number of rows and columns of seeds dataset:
(209, 1)
```

3. Write a Python program to get the number of observations, missing values and nan values.

CODE AND OUTPUT:

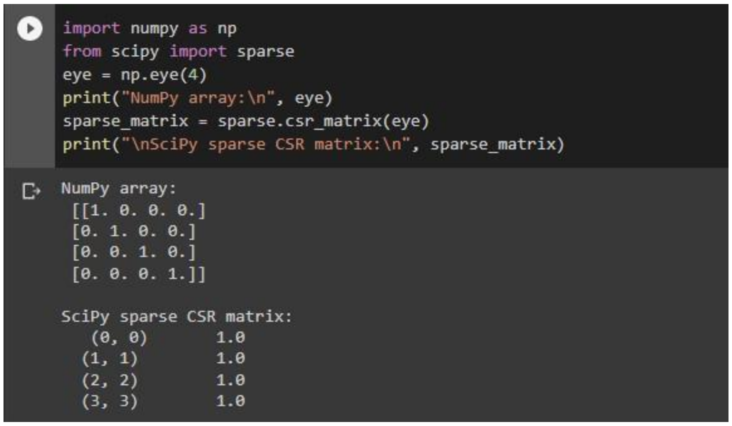


```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209 entries, 0 to 208
Data columns (total 1 columns):
 #   Column                                            Non-Null Count  Dtype 
---  ------                                            --------------  ----- 
 0   15.26      14.84   0.871   5.763   3.312   2.221   5.22    1  209 non-null    object
dtypes: object(1)
memory usage: 1.8+ KB
None
```

4. Write a Python program to create a 2-D array with ones on the diagonal and zeros elsewhere. Now convert the NumPy array to a SciPy sparse matrix in CSR format.

CODE AND OUTPUT:

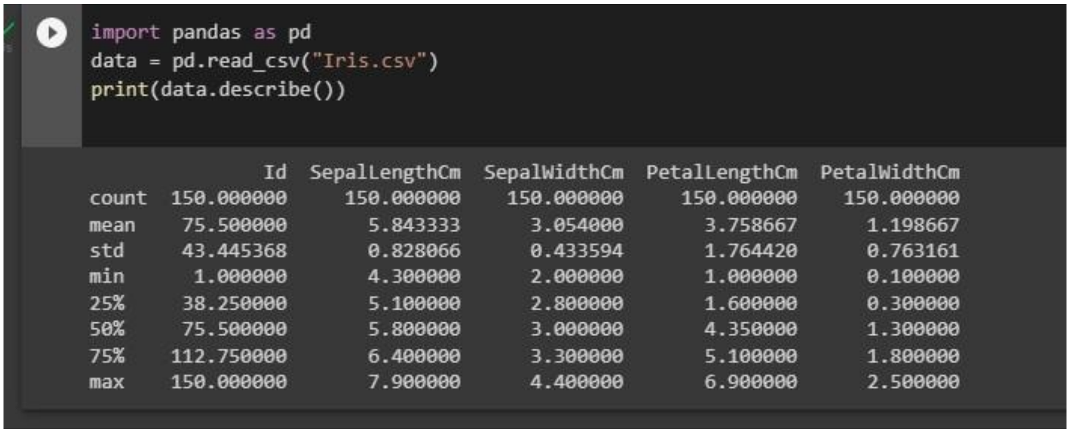


```
import numpy as np
from scipy import sparse
eye = np.eye(4)
print("NumPy array:\n", eye)
sparse_matrix = sparse.csr_matrix(eye)
print("\nSciPy sparse CSR matrix:\n", sparse_matrix)
```

```
NumPy array:
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]

SciPy sparse CSR matrix:
   (0, 0)	1.0
  (1, 1)	1.0
  (2, 2)	1.0
  (3, 3)	1.0
```

5. Write a Python program to view basic statistical details like percentile, mean, std etc. of iris data.

CODE AND OUTPUT:

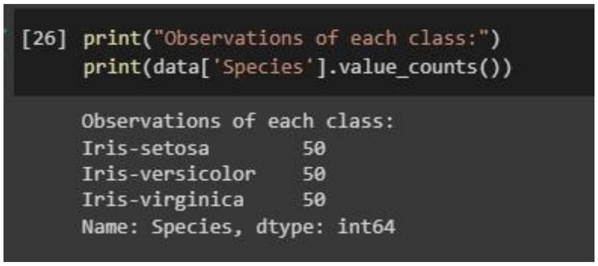


```
import pandas as pd
data = pd.read_csv("Iris.csv")
print(data.describe())
```

```
               Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count  150.000000     150.000000    150.000000     150.000000    150.000000
mean    75.500000       5.843333      3.054000       3.758667      1.198667
std     43.445368       0.828066      0.433594       1.764420      0.763161
min      1.000000       4.300000      2.000000       1.000000      0.100000
25%     38.250000       5.100000      2.800000       1.600000      0.300000
50%     75.500000       5.800000      3.000000       4.350000      1.300000
75%    112.750000       6.400000      3.300000       5.100000      1.800000
max    150.000000       7.900000      4.400000       6.900000      2.500000
```

6. Write a Python program to get observations of each class.

CODE AND OUTPUT:

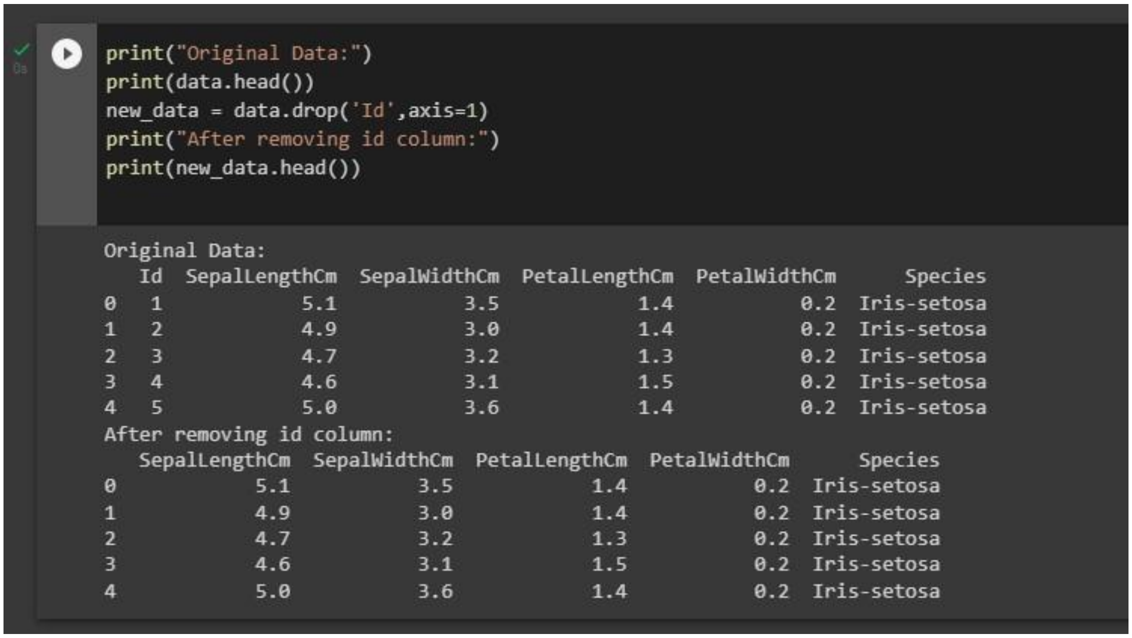


```
[26] print("Observations of each class:")
     print(data['Species'].value_counts())
```

```
Observations of each class:
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

7. Write a Python program to drop Id column from a given Dataframe and print the modified part. Call the csv to create the Dataframe.

CODE AND OUTPUT:

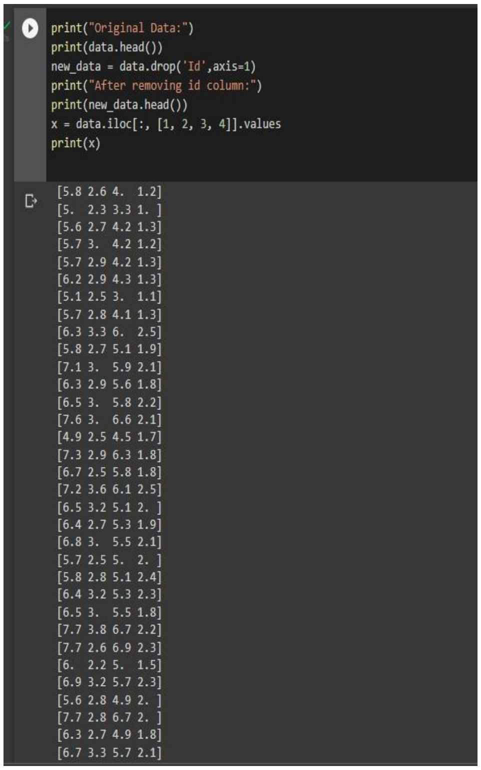


```
print("Original Data:")
print(data.head())
new_data = data.drop('Id',axis=1)
print("After removing id column:")
print(new_data.head())
```

```
Original Data:
   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
After removing id column:
   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
```

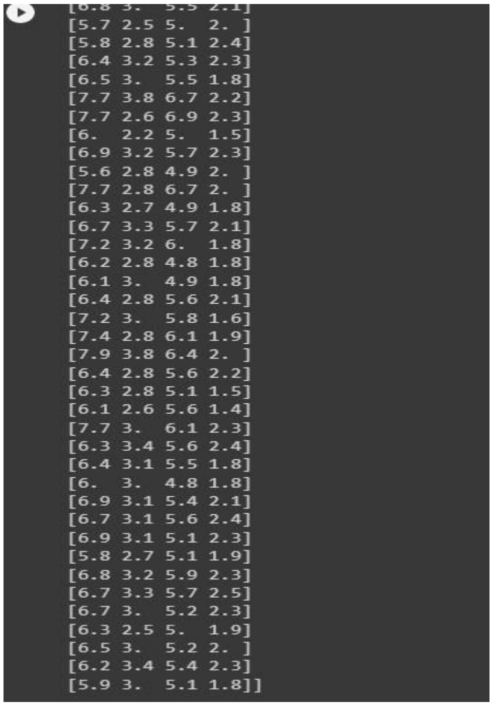
8. Write a Python program to access first four cells from a given Dataframe using the index and column labels. Call the csv to create the Dataframe.

CODE AND OUTPUT:



```
print("Original Data:")
print(data.head())
new_data = data.drop('Id',axis=1)
print("After removing id column:")
print(new_data.head())
x = data.iloc[:, [1, 2, 3, 4]].values
print(x)
```

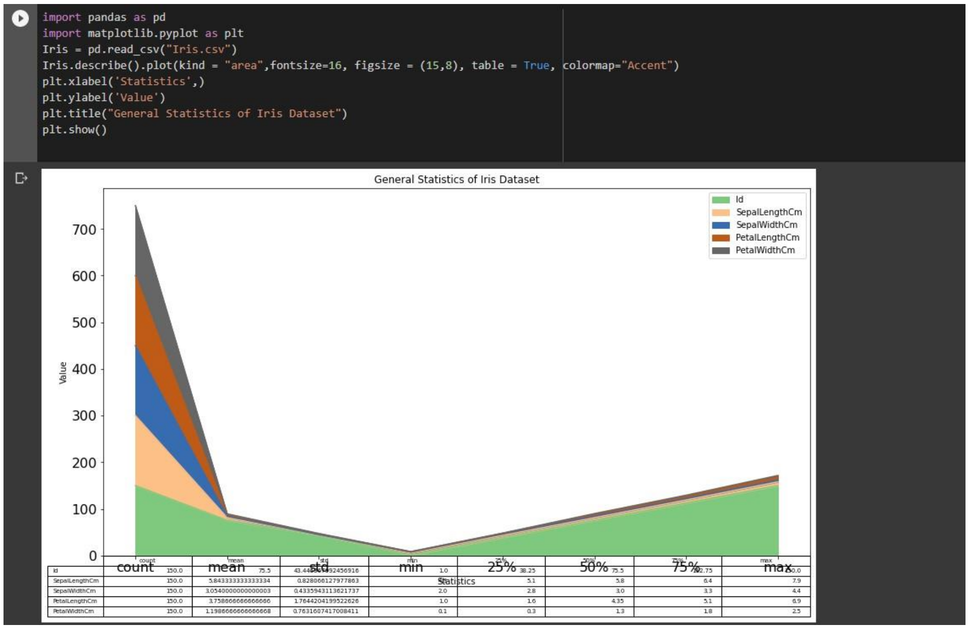
```
 [5.8 2.6 4.  1.2]
 [5.  2.3 3.3 1. ]
 [5.6 2.7 4.2 1.3]
 [5.7 3.  4.2 1.2]
 [5.7 2.9 4.2 1.3]
 [6.2 2.9 4.3 1.3]
 [5.1 2.5 3.  1.1]
 [5.7 2.8 4.1 1.3]
 [6.3 3.3 6.  2.5]
 [5.8 2.7 5.1 1.9]
 [7.1 3.  5.9 2.1]
 [6.3 2.9 5.6 1.8]
 [6.5 3.  5.8 2.2]
 [7.6 3.  6.6 2.1]
 [4.9 2.5 4.5 1.7]
 [7.3 2.9 6.3 1.8]
 [6.7 2.5 5.8 1.8]
 [7.2 3.6 6.1 2.5]
 [6.5 3.2 5.1 2. ]
 [6.4 2.7 5.3 1.9]
 [6.8 3.  5.5 2.1]
 [5.7 2.5 5.  2. ]
 [5.8 2.8 5.1 2.4]
 [6.4 3.2 5.3 2.3]
 [6.5 3.  5.5 1.8]
 [7.7 3.8 6.7 2.2]
 [7.7 2.6 6.9 2.3]
 [6.  2.2 5.  1.5]
 [6.9 3.2 5.7 2.3]
 [5.6 2.8 4.9 2. ]
 [7.7 2.8 6.7 2. ]
 [6.3 2.7 4.9 1.8]
 [6.7 3.3 5.7 2.1]
```



```
 [6.8 3.  5.5 2.1]
 [5.7 2.5 5.  2. ]
 [5.8 2.8 5.1 2.4]
 [6.4 3.2 5.3 2.3]
 [6.5 3.  5.5 1.8]
 [7.7 3.8 6.7 2.2]
 [7.7 2.6 6.9 2.3]
 [6.  2.2 5.  1.5]
 [6.9 3.2 5.7 2.3]
 [5.6 2.8 4.9 2. ]
 [7.7 2.8 6.7 2. ]
 [6.3 2.7 4.9 1.8]
 [6.7 3.3 5.7 2.1]
 [7.2 3.2 6.  1.8]
 [6.2 2.8 4.8 1.8]
 [6.1 3.  4.9 1.8]
 [6.4 2.8 5.6 2.1]
 [7.2 3.  5.8 1.6]
 [7.4 2.8 6.1 1.9]
 [7.9 3.8 6.4 2. ]
 [6.4 2.8 5.6 2.2]
 [6.3 2.8 5.1 1.5]
 [6.1 2.6 5.6 1.4]
 [7.7 3.  6.1 2.3]
 [6.3 3.4 5.6 2.4]
 [6.4 3.1 5.5 1.8]
 [6.  3.  4.8 1.8]
 [6.9 3.1 5.4 2.1]
 [6.7 3.1 5.6 2.4]
 [6.9 3.1 5.1 2.3]
 [5.8 2.7 5.1 1.9]
 [6.8 3.2 5.9 2.3]
 [6.7 3.3 5.7 2.5]
 [6.7 3.  5.2 2.3]
 [6.3 2.5 5.  1.9]
 [6.5 3.  5.2 2. ]
 [6.2 3.4 5.4 2.3]
 [5.9 3.  5.1 1.8]]
```

9. Write a Python program to create a plot to get a general Statistics of dataset

CODE AND OUTPUT:

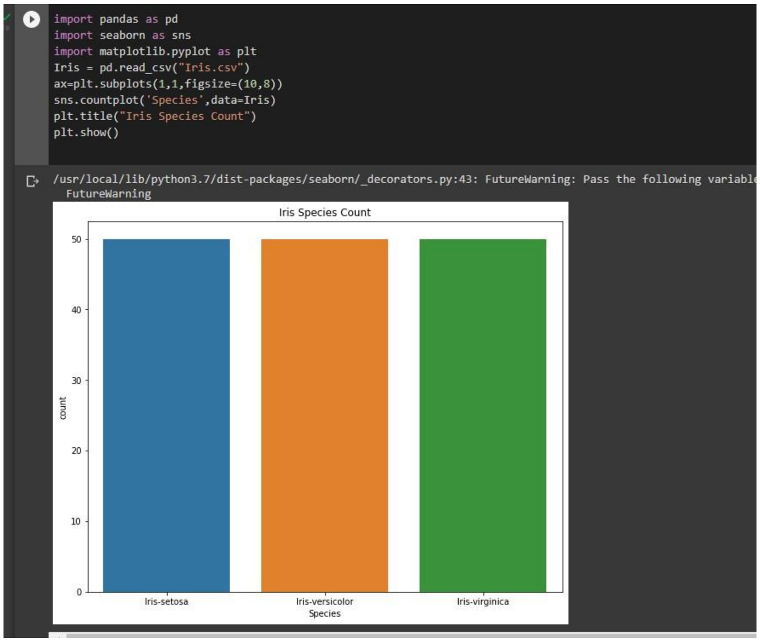


```
import pandas as pd
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
Iris.describe().plot(kind = "area",fontsize=16, figsize = (15,8), table = True, colormap="Accent")
plt.xlabel('Statistics',)
plt.ylabel('Value')
plt.title("General Statistics of Iris Dataset")
plt.show()
```

	count	mean	std	min	25%	50%	75%	max
Id	150.0	75.5	43.44[illegible]92456916	1.0	38.25	75.5	[illegible]2.75	[illegible]0.0
SepalLengthCm	150.0	5.843333333333334	0.828066127977863	[illegible]	5.1	5.8	6.4	7.9
SepalWidthCm	150.0	3.0540000000000003	0.4335943113621737	2.0	2.8	3.0	3.3	4.4
PetalLengthCm	150.0	3.758666666666666	1.7644204199522626	1.0	1.6	4.35	5.1	6.9
PetalWidthCm	150.0	1.1986666666666668	0.7631607417008411	0.1	0.3	1.3	1.8	2.5

10. Write a Python program to create a Bar plot to get the frequency of the three classes

CODE AND OUTPUT:

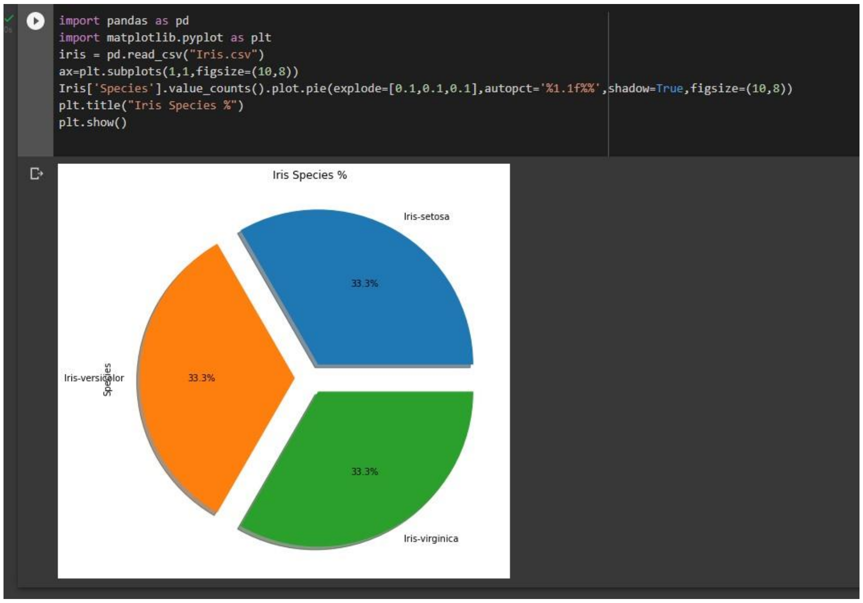


```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
ax=plt.subplots(1,1,figsize=(10,8))
sns.countplot('Species',data=Iris)
plt.title("Iris Species Count")
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable
  FutureWarning
```

11. Write a Python program to create a Pie plot to get the frequency of the three classes

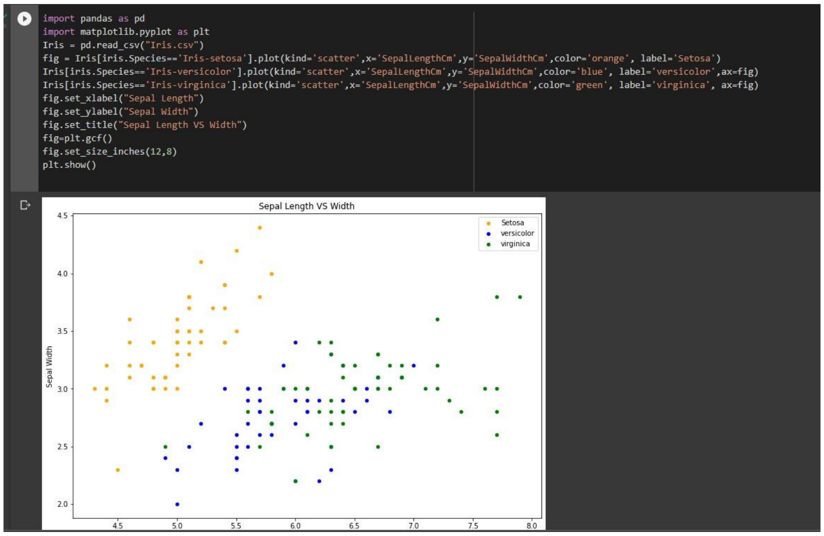
CODE AND OUTPUT:



```
import pandas as pd
import matplotlib.pyplot as plt
iris = pd.read_csv("Iris.csv")
ax=plt.subplots(1,1,figsize=(10,8))
Iris['Species'].value_counts().plot.pie(explode=[0.1,0.1,0.1],autopct='%1.1f%%',shadow=True,figsize=(10,8))
plt.title("Iris Species %")
plt.show()
```

12. Write a Python program to create a graph to find relationship between the length and width of the kernel.

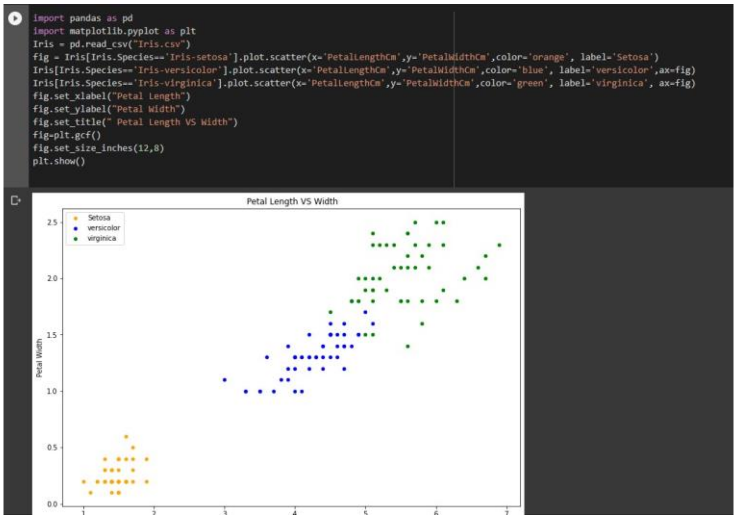
CODE AND OUTPUT:



```
import pandas as pd
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
fig = Iris[iris.Species=='Iris-setosa'].plot(kind='scatter',x='SepalLengthCm',y='SepalWidthCm',color='orange', label='Setosa')
Iris[iris.Species=='Iris-versicolor'].plot(kind='scatter',x='SepalLengthCm',y='SepalWidthCm',color='blue', label='versicolor',ax=fig)
Iris[iris.Species=='Iris-virginica'].plot(kind='scatter',x='SepalLengthCm',y='SepalWidthCm',color='green', label='virginica', ax=fig)
fig.set_xlabel("Sepal Length")
fig.set_ylabel("Sepal Width")
fig.set_title("Sepal Length VS Width")
fig=plt.gcf()
fig.set_size_inches(12,8)
plt.show()
```

13.Write a Python program to create a graph to find relationship between the perimeter and compactness.

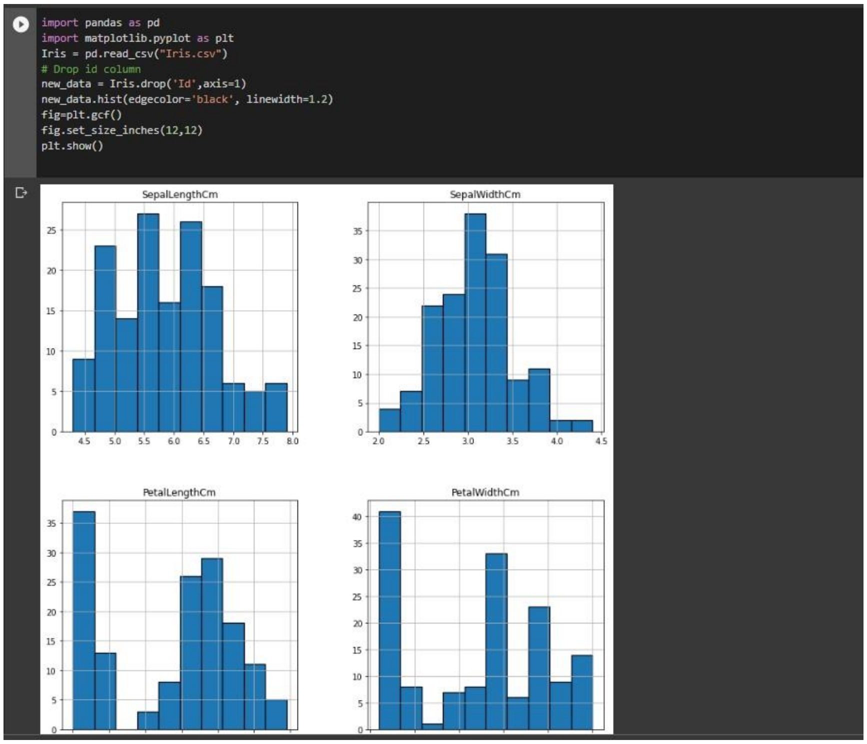
CODE AND OUTPUT:



```
import pandas as pd
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
fig = Iris[Iris.Species=='Iris-setosa'].plot.scatter(x='PetalLengthCm',y='PetalWidthCm',color='orange', label='Setosa')
Iris[Iris.Species=='Iris-versicolor'].plot.scatter(x='PetalLengthCm',y='PetalWidthCm',color='blue', label='versicolor',ax=fig)
Iris[Iris.Species=='Iris-virginica'].plot.scatter(x='PetalLengthCm',y='PetalWidthCm',color='green', label='virginica', ax=fig)
fig.set_xlabel("Petal Length")
fig.set_ylabel("Petal Width")
fig.set_title(" Petal Length VS Width")
fig=plt.gcf()
fig.set_size_inches(12,8)
plt.show()
```

14.	Write a Python program to create a graph to see how the length and width of are distributed.

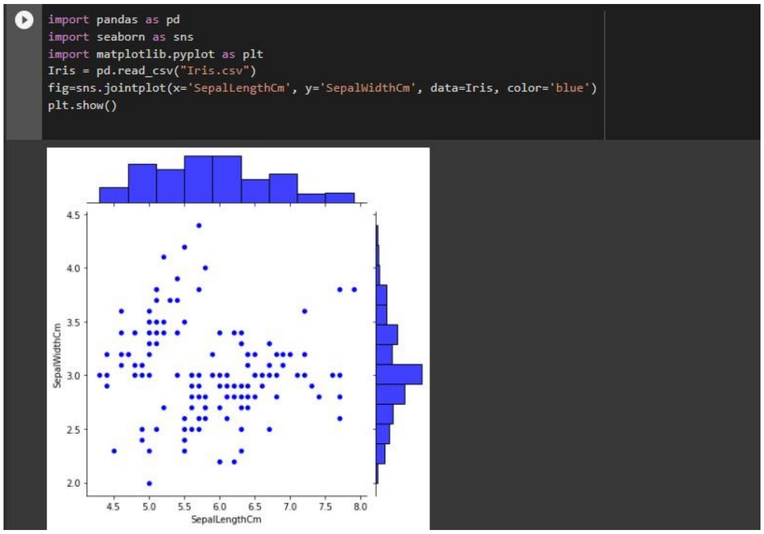
CODE AND OUTPUT:



```
import pandas as pd
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
# Drop id column
new_data = Iris.drop('Id',axis=1)
new_data.hist(edgecolor='black', linewidth=1.2)
fig=plt.gcf()
fig.set_size_inches(12,12)
plt.show()
```

15. Write a Python program to create a joinplot to describe individual distributions on the same plot between length and width .Note: joinplot - Draw a plot of two variables with bivariate and univariate graphs.

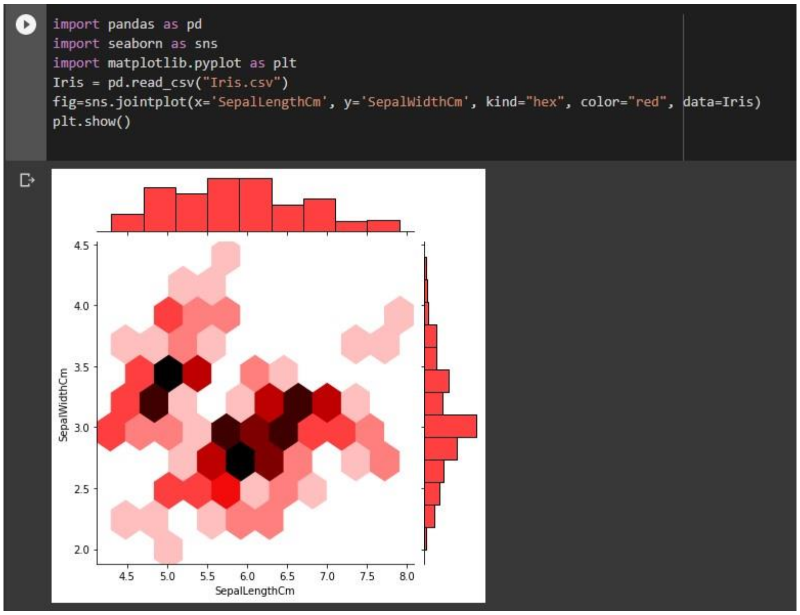
CODE AND OUTPUT:



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
fig=sns.jointplot(x='SepalLengthCm', y='SepalWidthCm', data=Iris, color='blue')
plt.show()
```

16. Write a Python program to create a joinplot using "hexbin" to describe individual distributions on the same plot between length and width.

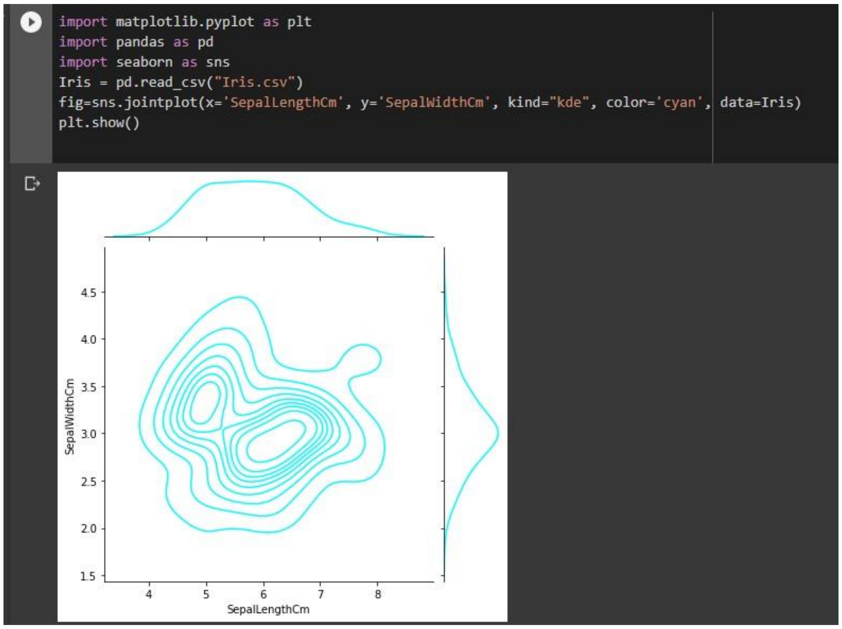
CODE AND OUTPUT:



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
fig=sns.jointplot(x='SepalLengthCm', y='SepalWidthCm', kind="hex", color="red", data=Iris)
plt.show()
```

17. Write a Python program to create a joinplot using "kde" to describe individual distributions on the same plot between length and width.

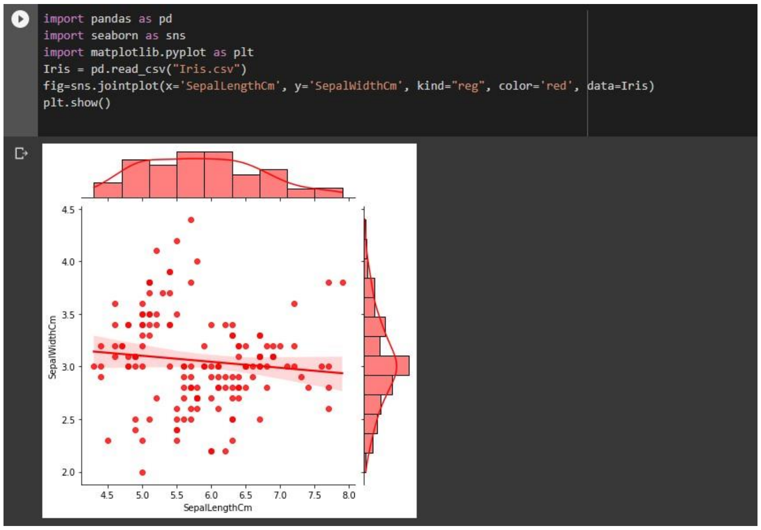
CODE AND OUTPUT:



```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
Iris = pd.read_csv("Iris.csv")
fig=sns.jointplot(x='SepalLengthCm', y='SepalWidthCm', kind="kde", color='cyan', data=Iris)
plt.show()
```

18. Write a Python program to create a joinplot and add regression and kernel density fits using "reg" to describe individual distributions on the same plot between length and width.

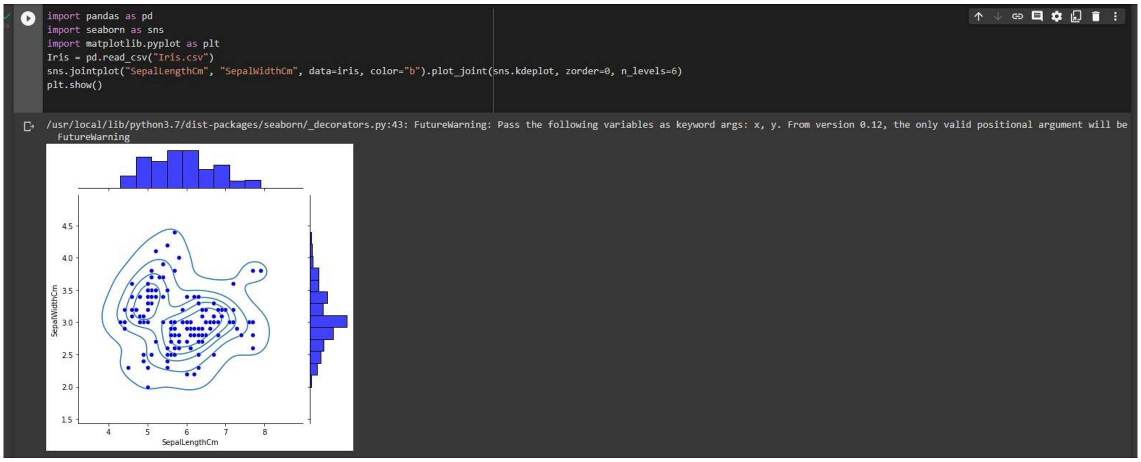
CODE AND OUTPUT:



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
fig=sns.jointplot(x='SepalLengthCm', y='SepalWidthCm', kind="reg", color='red', data=Iris)
plt.show()
```

19. Write a Python program to draw a scatterplot, then add a joint density estimate to describe individual distributions on the same plot between Length and width of the kernel

CODE AND OUTPUT:

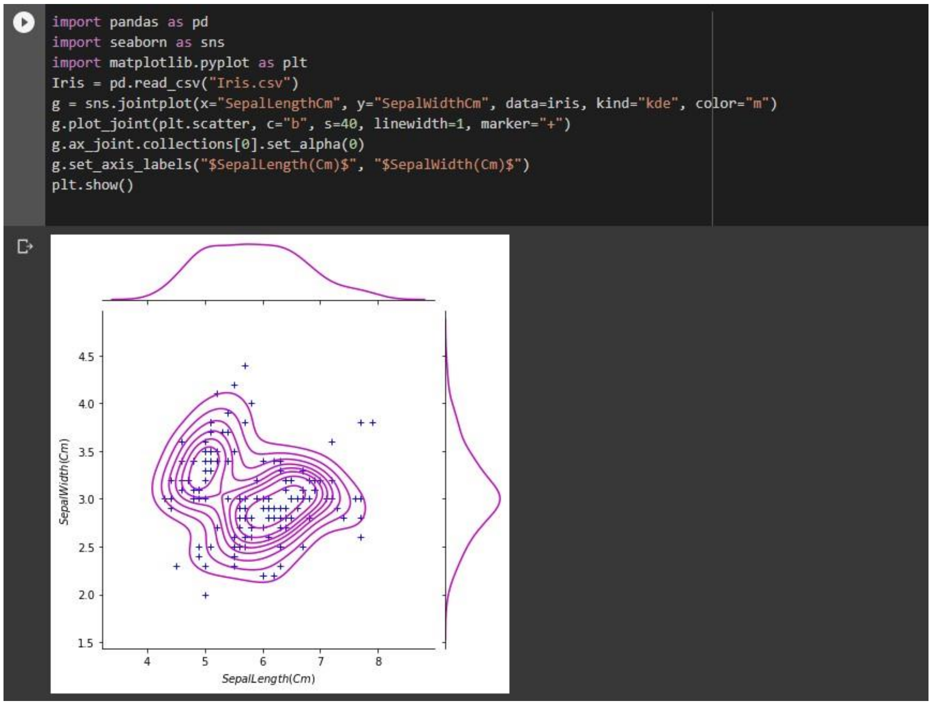


```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
sns.jointplot("SepalLengthCm", "SepalWidthCm", data=iris, color="b").plot_joint(sns.kdeplot, zorder=0, n_levels=6)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be
  FutureWarning
```

20. Write a Python program to create a joinplot using "kde" to describe individual distributions on the same plot between Length and width of the kernel and use '+' sign as marker.

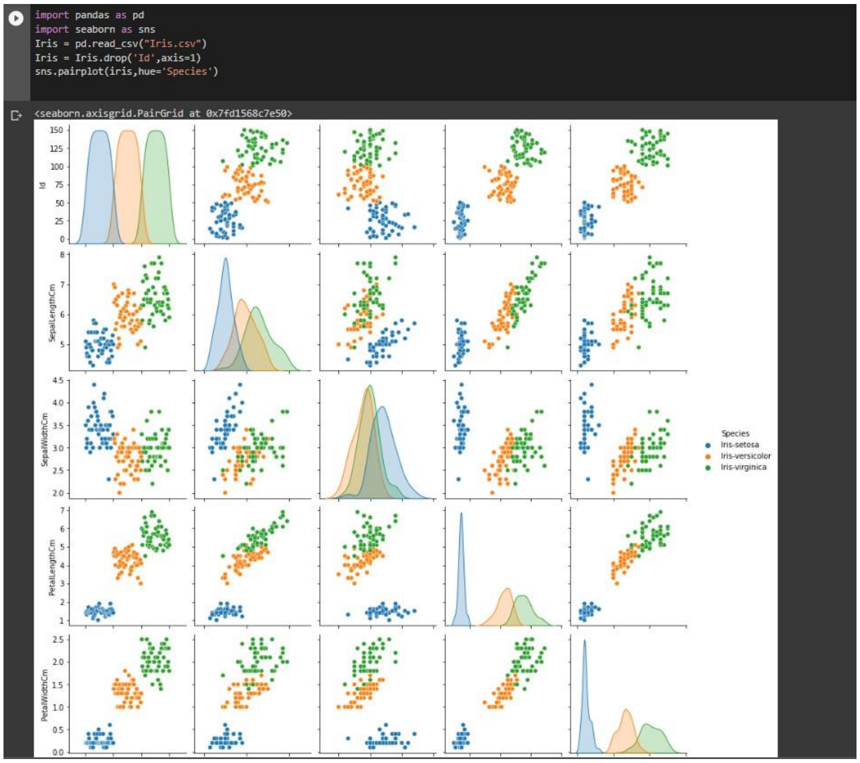
CODE AND OUTPUT:



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
Iris = pd.read_csv("Iris.csv")
g = sns.jointplot(x="SepalLengthCm", y="SepalWidthCm", data=iris, kind="kde", color="m")
g.plot_joint(plt.scatter, c="b", s=40, linewidth=1, marker="+")
g.ax_joint.collections[0].set_alpha(0)
g.set_axis_labels("$SepalLength(Cm)$", "$SepalWidth(Cm)$")
plt.show()
```

21. Write a Python program to create a pairplot of the data set and check which class seems to be the most separable.

CODE AND OUTPUT:

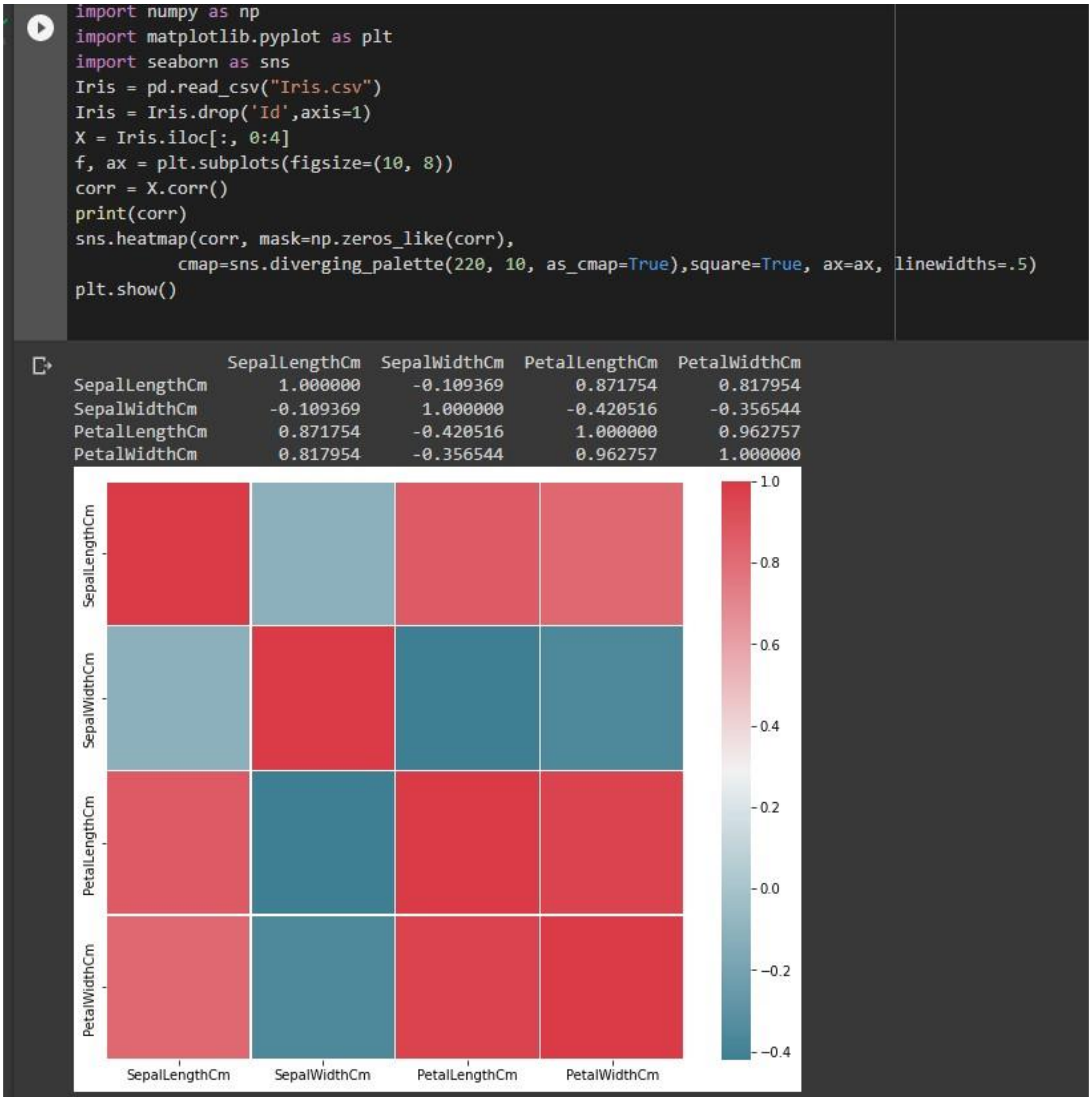


```
import pandas as pd
import seaborn as sns
Iris = pd.read_csv("Iris.csv")
Iris = Iris.drop('Id',axis=1)
sns.pairplot(iris,hue='Species')
```

```
<seaborn.axisgrid.PairGrid at 0x7fd1568c7e50>
```

22. Write a Python program to find the correlation between variables of wheat seeds data. Also create a heatmap using Seaborn to present their relations.

CODE AND OUTPUT:

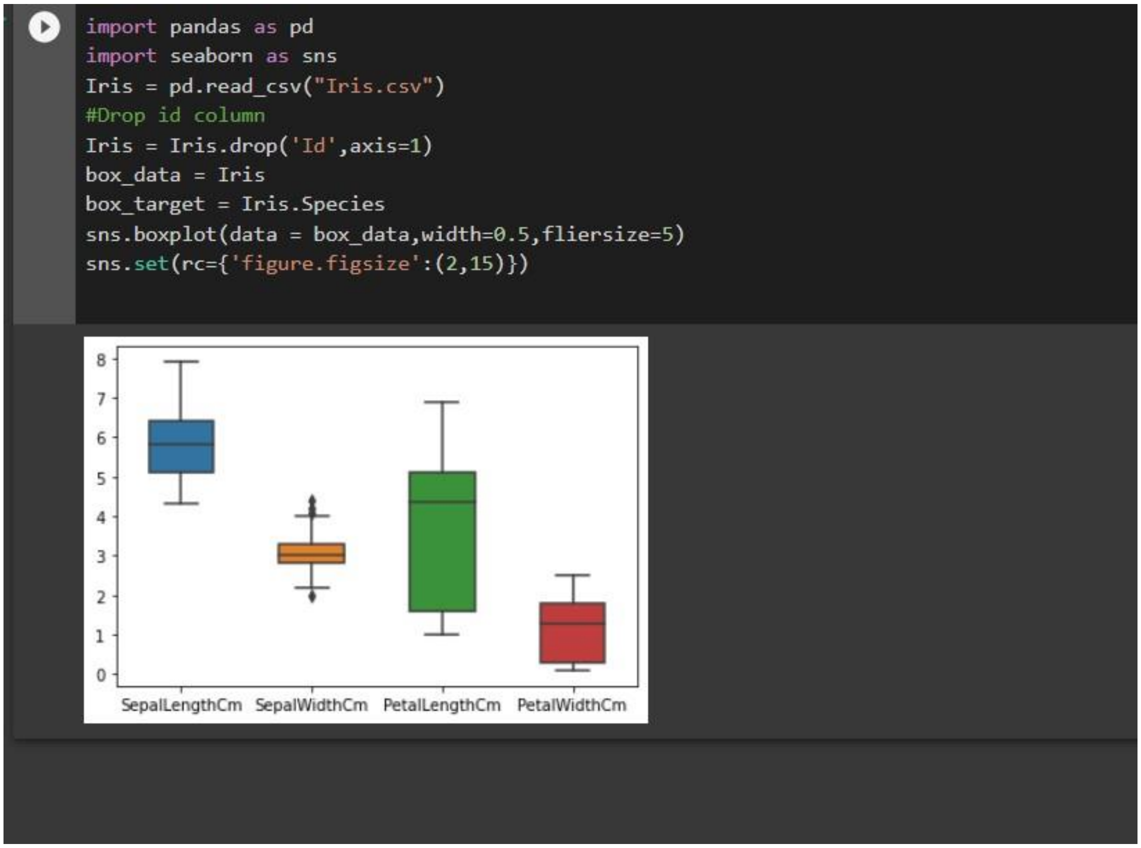


```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
Iris = pd.read_csv("Iris.csv")
Iris = Iris.drop('Id',axis=1)
X = Iris.iloc[:, 0:4]
f, ax = plt.subplots(figsize=(10, 8))
corr = X.corr()
print(corr)
sns.heatmap(corr, mask=np.zeros_like(corr),
          cmap=sns.diverging_palette(220, 10, as_cmap=True),square=True, ax=ax, linewidths=.5)
plt.show()
```

```
               SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
SepalLengthCm       1.000000     -0.109369       0.871754      0.817954
SepalWidthCm       -0.109369      1.000000      -0.420516     -0.356544
PetalLengthCm       0.871754     -0.420516       1.000000      0.962757
PetalWidthCm        0.817954     -0.356544       0.962757      1.000000
```

23. Write a Python program to create a box plot (or box-andwhisker plot) which shows the distribution of quantitative data in a way that facilitates comparisons between variables or across levels of a categorical variable of iris dataset. Use sEABORN

CODE AND OUTPUT:



```
import pandas as pd
import seaborn as sns
Iris = pd.read_csv("Iris.csv")
#Drop id column
Iris = Iris.drop('Id',axis=1)
box_data = Iris
box_target = Iris.Species
sns.boxplot(data = box_data,width=0.5,fliersize=5)
sns.set(rc={'figure.figsize':(2,15)})
```