Practical-1

Aim: Perform Exploratory Data Analysis on the given dataset using Python.

Procedure:

1. Import the dataset
2. View the head of the data
3. View the basic information of data and description of data
4. Find the unique value of data and verify the duplication of data
5. Plot a graph for unique value of dataset
6. Verify the presence of null value and replace the null value
7. Visualize the needed data

Program:

```
#Load the required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
#Load the data
df = pd.read_csv("C://Users//acrop//Downloads//archive//tested.csv")
```

```
#View the data
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

```
#Describe the data

df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	0.363636	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.481622	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	0.000000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	0.000000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	1.000000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	1.000000	3.000000	76.000000	8.000000	9.000000	512.329200

```
#unique values

df['Pclass'].unique()
```

```
array([3, 2, 1], dtype=int64)
```

```
df['Survived'].unique()
```

```
array([0, 1], dtype=int64)
```

```
df['Sex'].unique()
```

```
array(['male', 'female'], dtype=object)
```

```
#checking duplicate values
df.nunique()
```

```
PassengerId    418
Survived         2
Pclass           3
Name           418
Sex              2
Age             80
SibSp            7
Parch            8
Ticket         363
Fare           170
Cabin           77
Embarked         3
dtype: int64
```

```
#Find null values or check for missing values

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

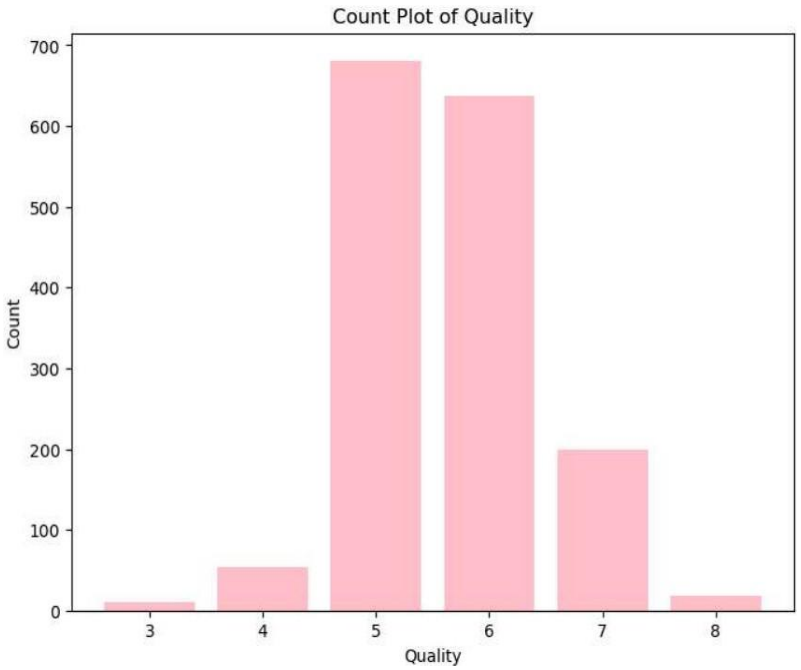
```
PassengerId       0
Survived          0
Pclass            0
Name              0
Sex               0
Age              86
SibSp             0
Parch             0
Ticket            0
Fare              1
Cabin           327
Embarked          0
dtype: int64
```

```
#Replace null values

df.replace(np.nan,'0',inplace = True)
```

```
#Check the changes now
df.isnull().sum()
```

```
PassengerId     0
Survived        0
Pclass          0
Name            0
Sex             0
Age             0
SibSp           0
Parch           0
Ticket          0
Fare            0
Cabin           0
Embarked        0
dtype: int64
```

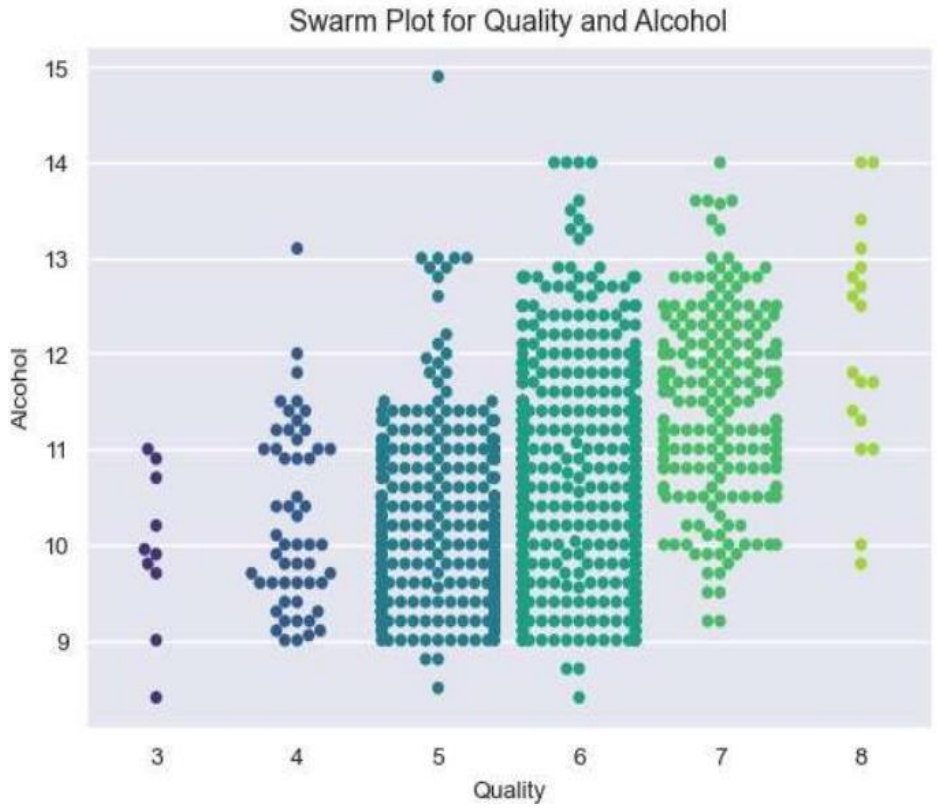
```
# Using Matplotlib to create a count plot
plt.figure(figsize=(8, 6))
plt.bar(quality_counts.index, quality_counts, color='PINK')
plt.title('Count Plot of Quality')
plt.xlabel('Quality')
plt.ylabel('Count')
plt.show()
```




```
#Swarm Plot

# Assuming 'df' is your DataFrame
plt.figure(figsize=(10, 8))
```

```
<Figure size 1000x800 with 0 Axes>

<Figure size 1000x800 with 0 Axes>
```

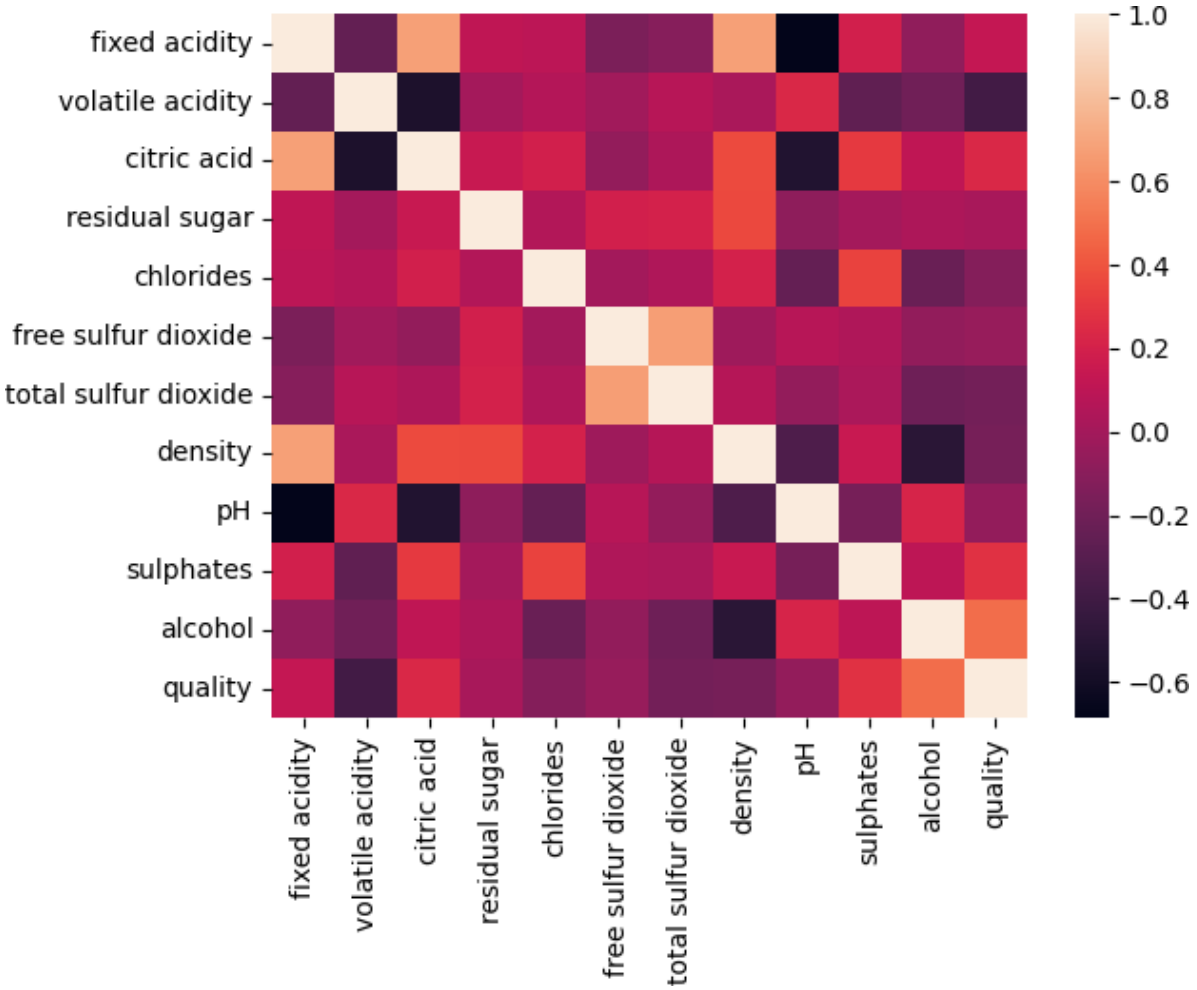
```
# Using Seaborn to create a swarm plot
sns.swarmplot(x="quality", y="alcohol", data=df, palette='viridis')

plt.title('Swarm Plot for Quality and Alcohol')
plt.xlabel('Quality')
plt.ylabel('Alcohol')
plt.show()
```




```
#pair Plot

#set the color palette
sns.set_palette("Pastel1")
plt.figure(figsize=(10,7))
sns.pairplot(df)
plt.suptitle('Pair Plot for Dataframe')
plt.show()
```


Pair Plot for Dataframe
fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality

```
#Heatmap
sns.heatmap(df.corr())
```



Practical-2

Aim: Calculate mean, median and mode of the first 50 records in the given dataset using python

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv("/content/winequality-red.csv")
```

```
df
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1591	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1592	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1593	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1594	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1595	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1596 rows × 12 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1596 entries, 0 to 1595
Data columns (total 12 columns):
 #    Column                Non-Null Count  Dtype  
---   ------                --------------  -----  
 0    fixed acidity         1596 non-null   float64
 1    volatile acidity      1596 non-null   float64
 2    citric acid           1596 non-null   float64
 3    residual sugar        1596 non-null   float64
 4    chlorides             1596 non-null   float64
 5    free sulfur dioxide   1596 non-null   float64
 6    total sulfur dioxide  1596 non-null   float64
 7    density               1596 non-null   float64
 8    pH                    1596 non-null   float64
 9    sulphates             1596 non-null   float64
 10   alcohol               1596 non-null   float64
 11   quality               1596 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 149.8 KB
```

```
meanQ=df["quality"].mean()
print(meanQ)
```

```
5.637218045112782
```

```
medianQ=df["quality"].median()
print(medianQ)
```

```
6.0
```

```
Mode=df["quality"].mode()
print(Mode)
```

```
0    5
Name: quality, dtype: int64
```