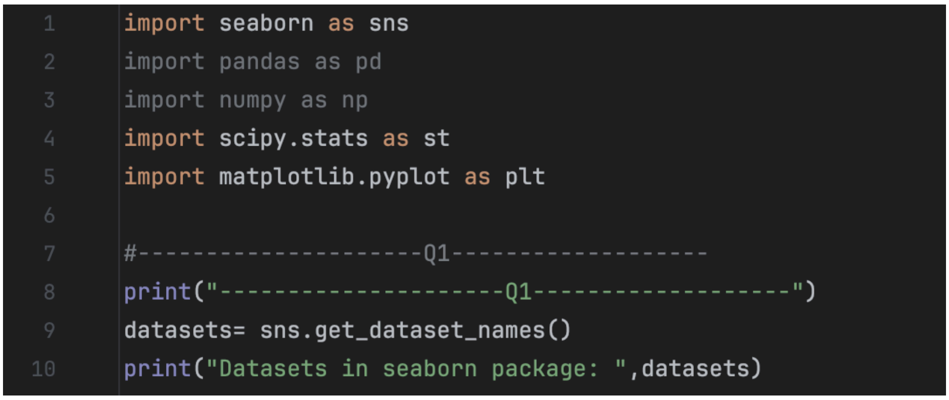
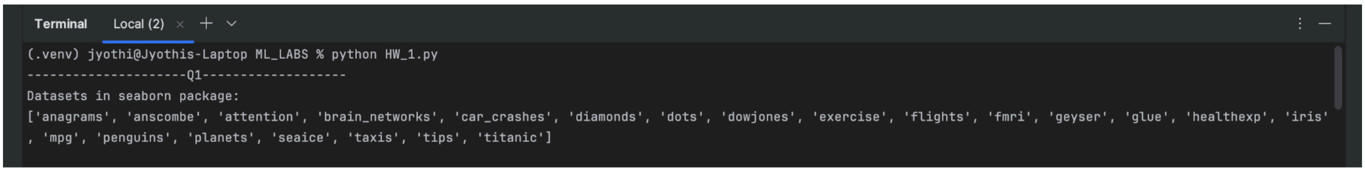
Q1.
Code:



```python
import seaborn as sns
import pandas as pd
import numpy as np
import scipy.stats as st
import matplotlib.pyplot as plt

#--------------------Q1------------------
print("--------------------Q1-------------------")
datasets= sns.get_dataset_names()
print("Datasets in seaborn package: ",datasets)
```

Output:



```
(.venv) jyothi@Jyothis-Laptop ML_LABS % python HW_1.py
--------------------Q1------------------
Datasets in seaborn package:
['anagrams', 'anscombe', 'attention', 'brain_networks', 'car_crashes', 'diamonds', 'dots', 'dowjones', 'exercise', 'flights', 'fmri', 'geyser', 'glue', 'healthexp', 'iris'
, 'mpg', 'penguins', 'planets', 'seaice', 'taxis', 'tips', 'titanic']
```

Q2.

Dataset title	**# of Observations**	**List of Categorical features**	**List of Numerical features**
diamonds	53940	['cut', 'color', 'clarity']	['carat', 'depth', 'table', 'price', 'x', 'y', 'z']
iris	150	['species']	['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
tips	244	['sex', 'smoker', 'day', 'time']	['total_bill', 'tip', 'size']
penguins	344	['species', 'island', 'sex']	['bill_length_mm', 'bill_depth_mm', 'flipper_length_mm', 'body_mass_g']
titanic	891	['survived', 'pclass','sex', 'embarked', 'class', 'who', 'deck', 'embark_town', 'alive',’alone’,’adult_male’]	['age', 'sibsp', 'parch', 'fare']

In diamonds dataset ,

Carat - Values (0.2 - 5.01) - Numerical feature

Cut - [Idea, Premium, Very Good, Good and Fair] - Categorical feature
Color - [D, E, F, G, H, I, and J]. - Categorical feature
Clarity - [SI1, VS2, SI2, VS1 , VVS2, VVS1, IF, I1] - Categorical feature
Depth - Value (43 - 79) - Numerical feature
Table - Value (43 - 95) - Numerical feature
Price - Value (326 - 18823) - Numerical feature
X - Length (0 - 10.74) - Numerical feature
Y - Width (0 - 58.9) - Numerical feature
Z - Height (0 - 31.8) - Numerical feature

In iris dataset ,

Sepal length - Values (4.3 to 7.9) - Numerical feature
Sepal width - Values (2.0 to 4.4) - Numerical feature
Petal length - Values (1.0 to 6.9) - Numerical feature
Petal width - Values (0.1 to 2.5) - Numerical feature
Species - [Setosa, Versicolor, Virginica]- Categorical feature

In tips dataset ,

Total bill - Values (3.07 - 50.81) - Numerical feature
Tip - Values (1.0 - 10.0) - Numerical feature
Sex - [Male or Female] - Categorical feature
Smoker - [Yes or No] - Categorical feature
Day - [Sat, Sun , Thur, Fri] - Categorical feature
Time - [Dinner, Lunch] - Categorical feature
Size - Number of people at who had dinner/lunch - Numerical feature

In penguins dataset ,

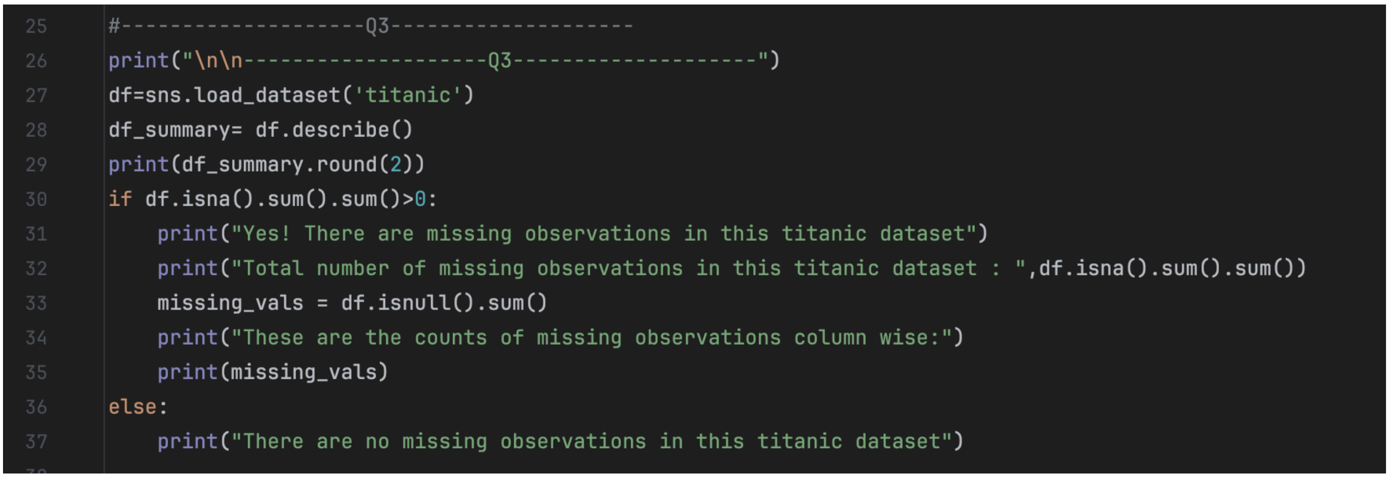
Bill length - Values (32.1 to 59.6) - Numerical feature
Bill depth - Values (13.1 to 21.5) - Numerical feature
Flipper length - Length ranges from 172.0 to 231.0 - Numerical feature
Body mass - Weight of the penguin (2700 - 6300) - Numerical feature
Sex - [Male or Female] - Categorical feature
Island - [Biscoe, Dream, Torgersen] - Categorical feature
Species - [Adelie, Gentoo, Chinstrap] - Categorical feature

In titanic dataset,

Survived - 0 or 1 - One hot encoded - Categorical feature
PClass - 1, 2, 3 - One hot encoded - Categorical feature
Sex - Male or Female - Categorical feature
Age - Age of the person traveling - (0.42 - 80) - Numerical feature
Sibsp - Number of siblings and spouse - (0.0 - 8.0) - Numerical feature
Parch - Number of parents and children - (0.0 - 6.0) - Numerical feature
Fare - Cost in dollars (0.0 to 512.32) - Numerical feature
Embarked - S, C, Q - Categorical feature
Class - First, second, third - Categorical feature
Who - Man, Woman, Child - Categorical feature
Adult_male - True or False - Categorical feature
Deck - A, B, C, D, E, F, G - Categorical feature

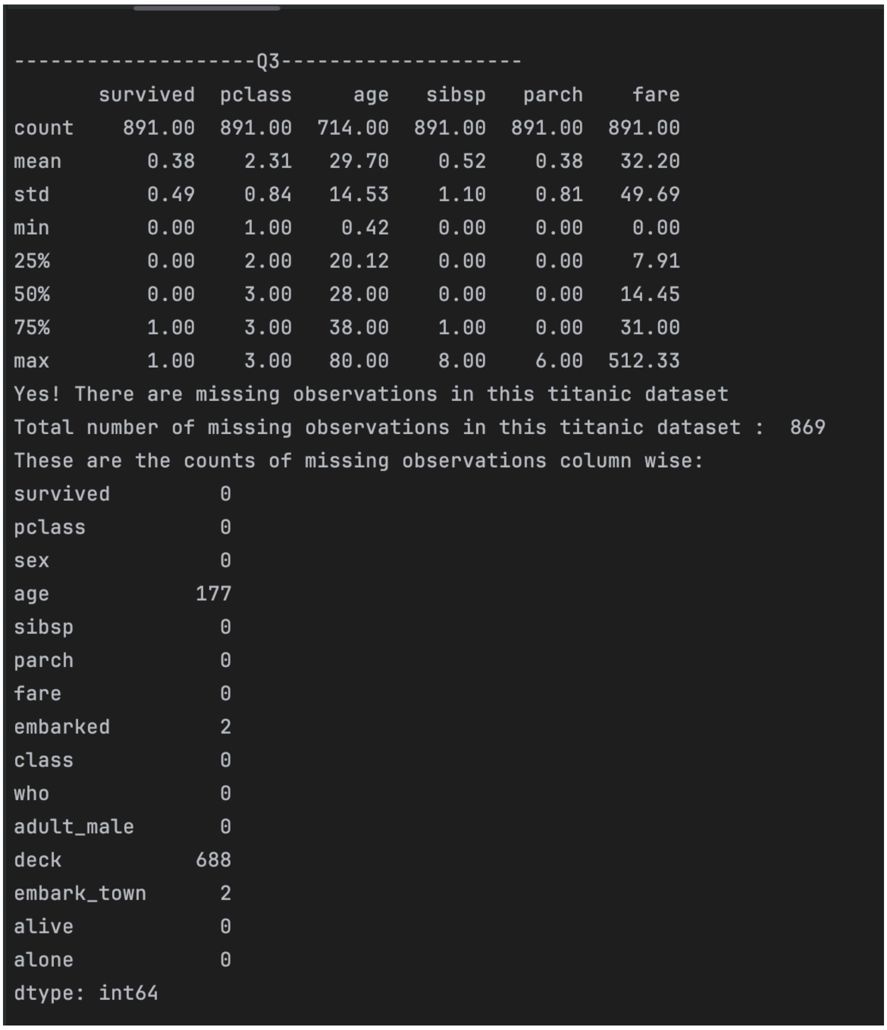
Embark town - Southampton, Cherbourg, Queenstown - Categorical feature
Alive - Yes or No - Categorical feature
Alone - True or False - Categorical feature

Q3.
Code:



```
#--------------------Q3--------------------
print("\n\n--------------------Q3--------------------")
df=sns.load_dataset('titanic')
df_summary= df.describe()
print(df_summary.round(2))
if df.isna().sum().sum()>0:
    print("Yes! There are missing observations in this titanic dataset")
    print("Total number of missing observations in this titanic dataset : ",df.isna().sum().sum())
    missing_vals = df.isnull().sum()
    print("These are the counts of missing observations column wise:")
    print(missing_vals)
else:
    print("There are no missing observations in this titanic dataset")
```

Output:

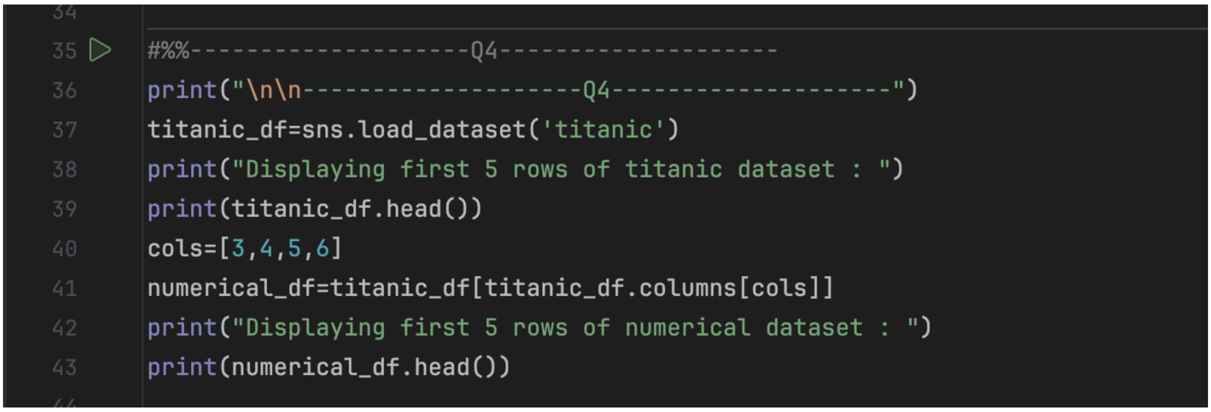


```
--------------------Q3--------------------
       survived  pclass     age   sibsp   parch    fare
count    891.00  891.00  714.00  891.00  891.00  891.00
mean       0.38    2.31   29.70    0.52    0.38   32.20
std        0.49    0.84   14.53    1.10    0.81   49.69
min        0.00    1.00    0.42    0.00    0.00    0.00
25%        0.00    2.00   20.12    0.00    0.00    7.91
50%        0.00    3.00   28.00    0.00    0.00   14.45
75%        1.00    3.00   38.00    1.00    0.00   31.00
max        1.00    3.00   80.00    8.00    6.00  512.33
Yes! There are missing observations in this titanic dataset
Total number of missing observations in this titanic dataset :  869
These are the counts of missing observations column wise:
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

Identifying whether the data type is nominal, ordinal, interval, ratio type:
Survived, Sex, Embarked, Who, Adult_Male, Deck, Embarked_Town, Alive and Alone are nominal data
PClass, Class is ordinal data
Age, fare are ratio type of data.

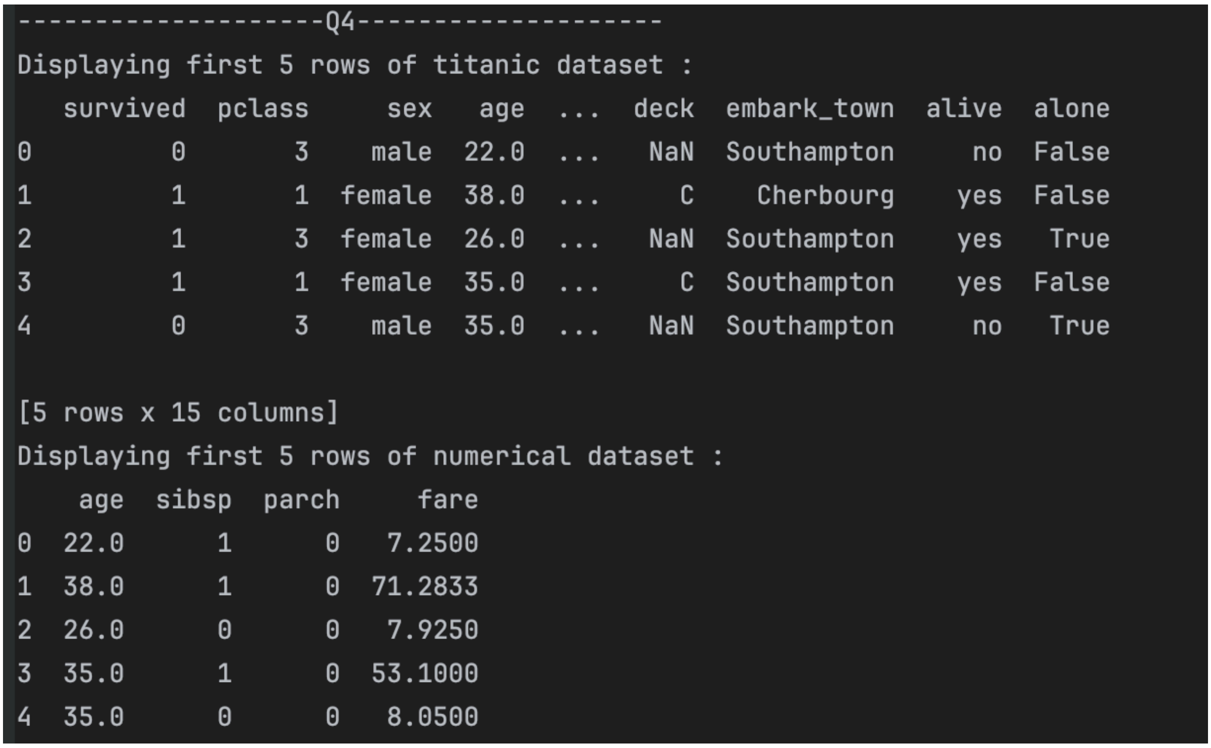
- 0.5 pts: parch and fare should also be ratio da

Q4.
Code:



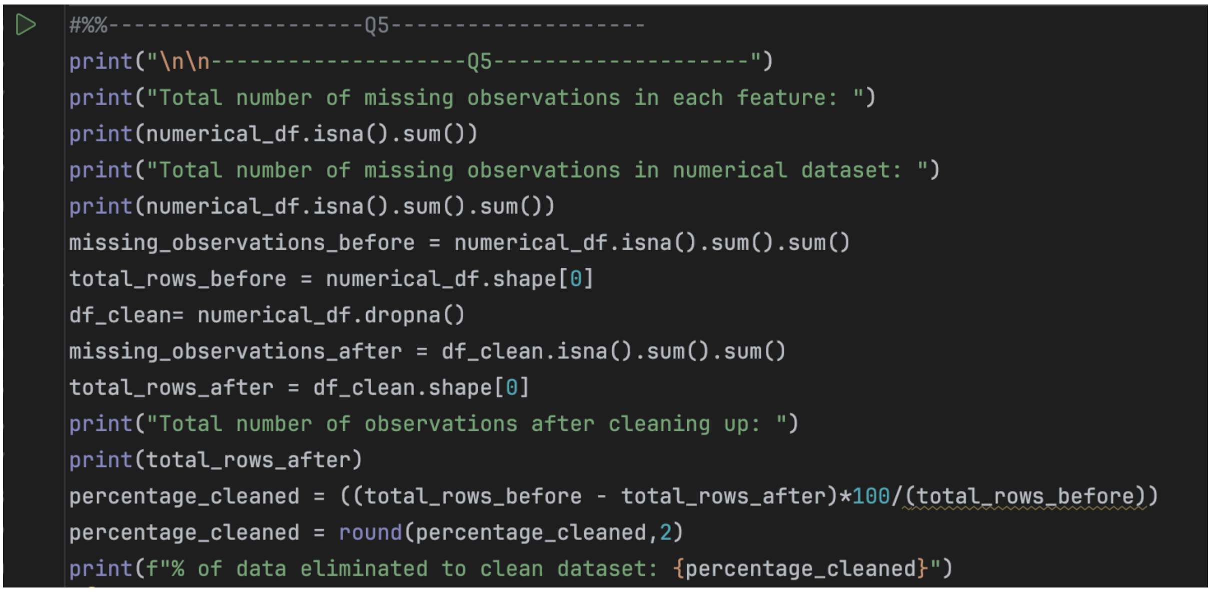
```
#%%------------------Q4------------------
print("\n\n------------------Q4-------------------")
titanic_df=sns.load_dataset('titanic')
print("Displaying first 5 rows of titanic dataset : ")
print(titanic_df.head())
cols=[3,4,5,6]
numerical_df=titanic_df[titanic_df.columns[cols]]
print("Displaying first 5 rows of numerical dataset : ")
print(numerical_df.head())
```

Output:



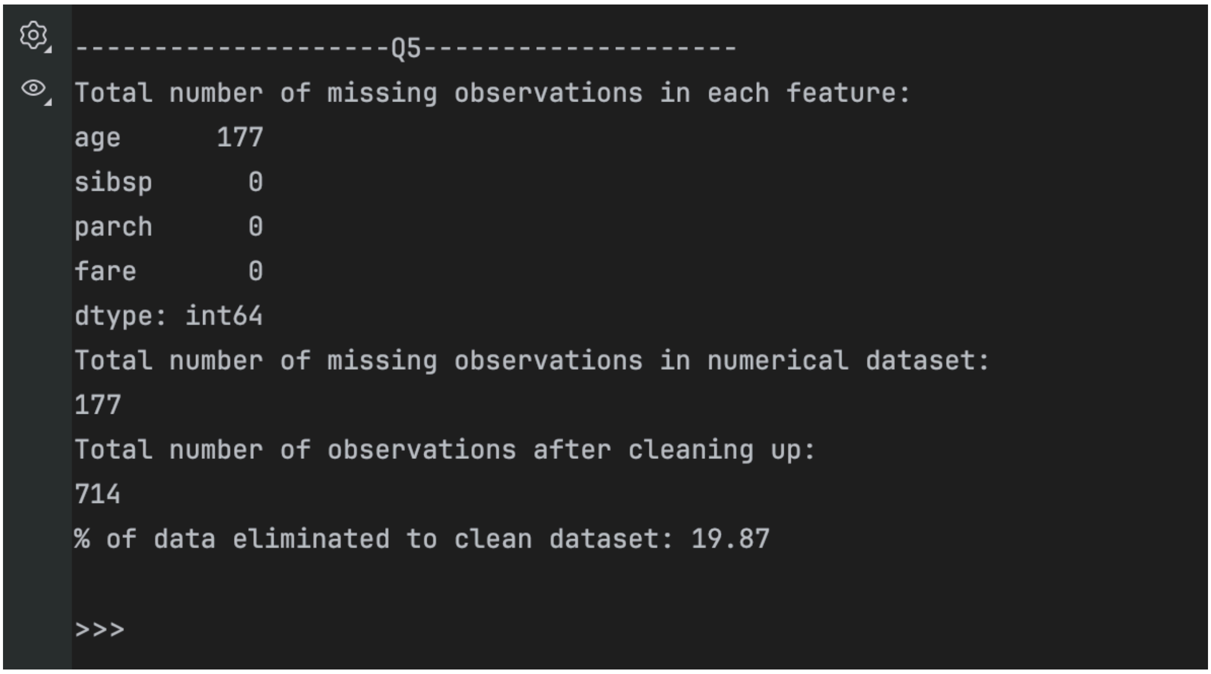
```
-------------------Q4-------------------
Displaying first 5 rows of titanic dataset :
   survived  pclass     sex   age  ...  deck  embark_town  alive  alone
0         0       3    male  22.0  ...   NaN  Southampton     no  False
1         1       1  female  38.0  ...     C    Cherbourg    yes  False
2         1       3  female  26.0  ...   NaN  Southampton    yes   True
3         1       1  female  35.0  ...     C  Southampton    yes  False
4         0       3    male  35.0  ...   NaN  Southampton     no   True

[5 rows x 15 columns]
Displaying first 5 rows of numerical dataset :
    age  sibsp  parch     fare
0  22.0      1      0   7.2500
1  38.0      1      0  71.2833
2  26.0      0      0   7.9250
3  35.0      1      0  53.1000
4  35.0      0      0   8.0500
```

Q5.
Code:



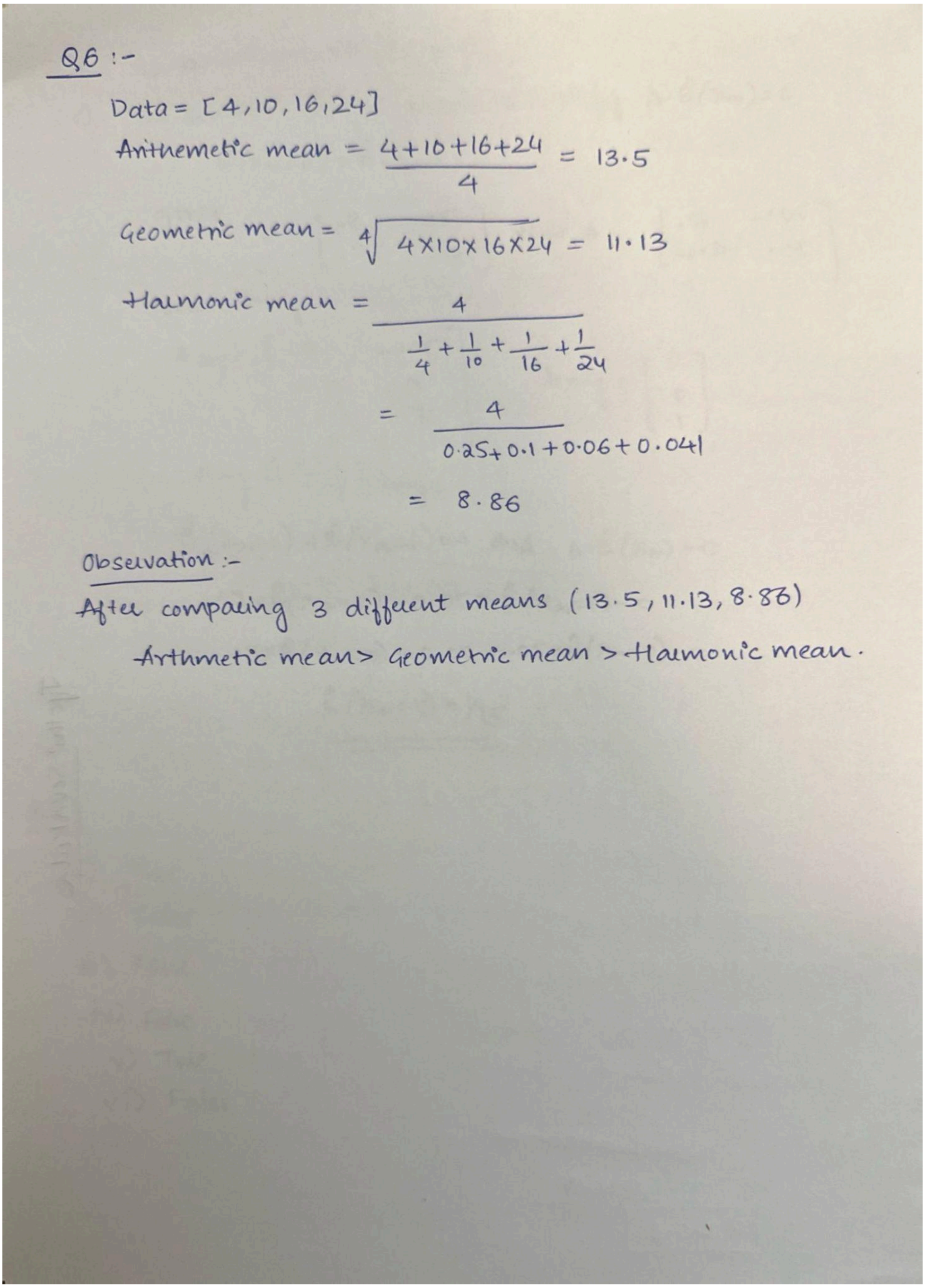
```
#%%-------------------Q5-------------------
print("\n\n-------------------Q5-------------------")
print("Total number of missing observations in each feature: ")
print(numerical_df.isna().sum())
print("Total number of missing observations in numerical dataset: ")
print(numerical_df.isna().sum().sum())
missing_observations_before = numerical_df.isna().sum().sum()
total_rows_before = numerical_df.shape[0]
df_clean= numerical_df.dropna()
missing_observations_after = df_clean.isna().sum().sum()
total_rows_after = df_clean.shape[0]
print("Total number of observations after cleaning up: ")
print(total_rows_after)
percentage_cleaned = ((total_rows_before - total_rows_after)*100/(total_rows_before))
percentage_cleaned = round(percentage_cleaned,2)
print(f"% of data eliminated to clean dataset: {percentage_cleaned}")
```

Output:



```
--------------------Q5--------------------
Total number of missing observations in each feature:
age      177
sibsp      0
parch      0
fare       0
dtype: int64
Total number of missing observations in numerical dataset:
177
Total number of observations after cleaning up:
714
% of data eliminated to clean dataset: 19.87


>>>
```

Q6.



Q6 :-

Data = [4, 10, 16, 24]

Arithemetic mean = $\frac{4+10+16+24}{4}$ = 13.5

Geometric mean = $\sqrt[4]{4 \times 10 \times 16 \times 24}$ = 11.13

Harmonic mean = $\frac{4}{\frac{1}{4}+\frac{1}{10}+\frac{1}{16}+\frac{1}{24}}$

= $\frac{4}{0.25+0.1+0.06+0.041}$

= 8.86

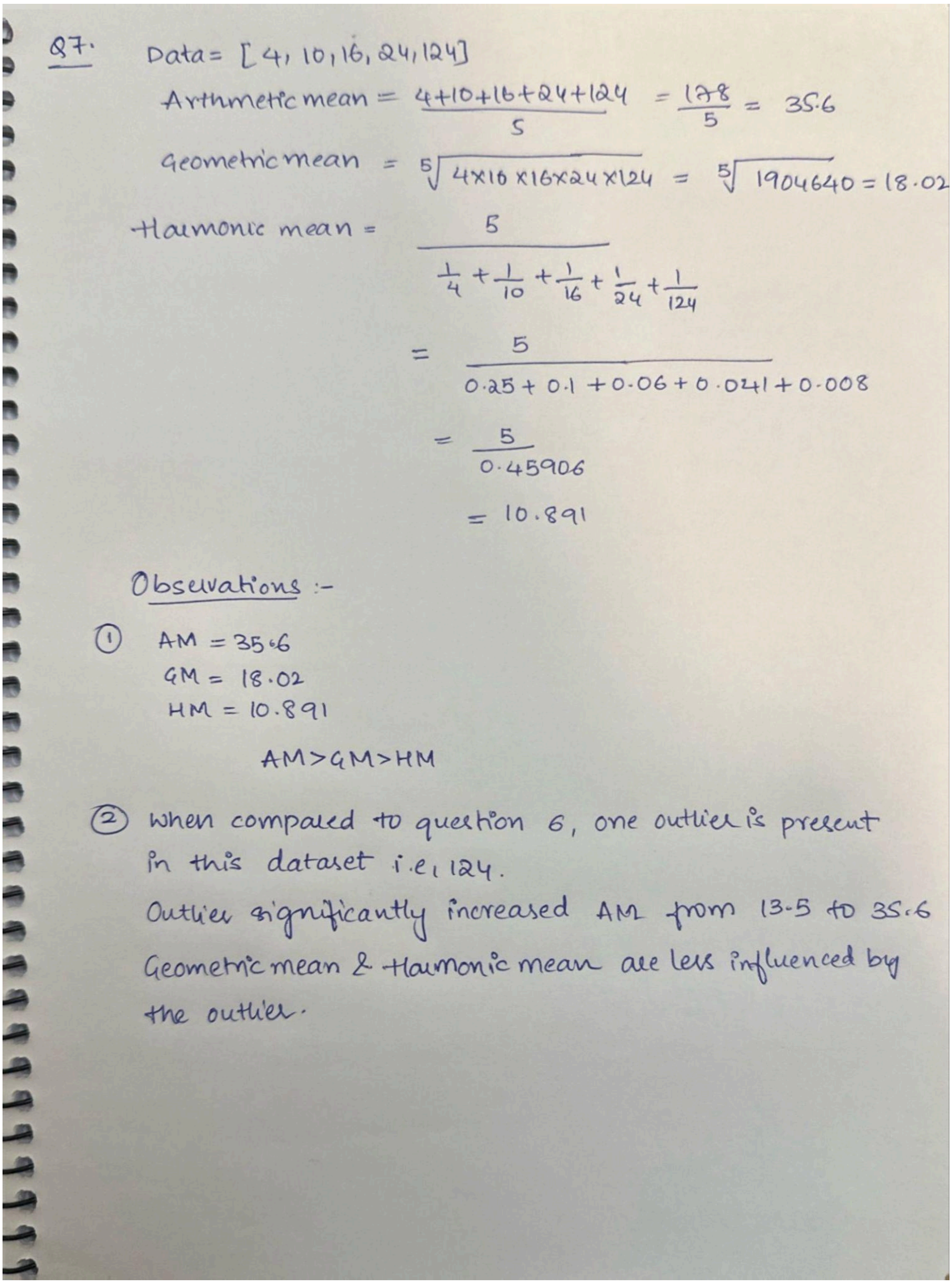
Observation :-

After comparing 3 different means (13.5, 11.13, 8.86)

Arthmetic mean > Geometric mean > Harmonic mean.

-1 pt: bigger picture here is that all the means are roughly the san

Q7.



Q7. Data = [4, 10, 16, 24, 124]

Arthmetic mean $= \frac{4+10+16+24+124}{5} = \frac{178}{5} = 35.6$

Geometric mean $= \sqrt[5]{4\times10\times16\times24\times124} = \sqrt[5]{1904640} = 18.02$

Harmonic mean $= \dfrac{5}{\frac{1}{4}+\frac{1}{10}+\frac{1}{16}+\frac{1}{24}+\frac{1}{124}}$

$= \dfrac{5}{0.25+0.1+0.06+0.041+0.008}$

$= \dfrac{5}{0.45906}$

$= 10.891$

Observations :-

① AM = 35.6
GM = 18.02
HM = 10.891

AM>GM>HM

② when compared to question 6, one outlier is present in this dataset i.e, 124.

Outlier significantly increased AM from 13.5 to 35.6

Geometric mean & Harmonic mean are less influenced by the outlier.

- 5 pts. : in class, it was stated that because of the 0's in the data, we can't calculate the values

Q8.

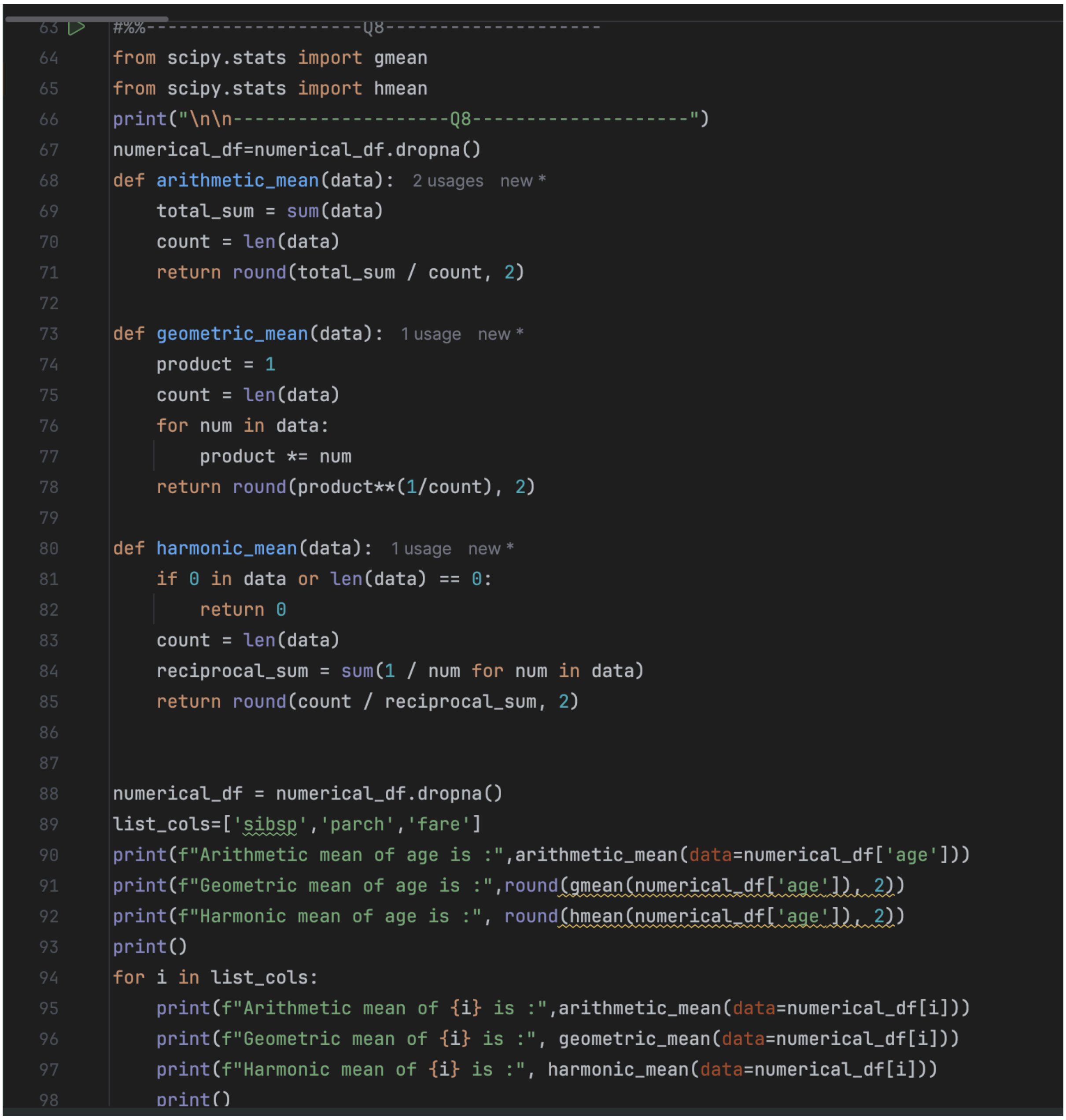
For age AM>GM>HM

For sibsp, AM=0.51 where as GM and Hm are 0 as it has zero in its observations

For parch,AM= 0.43 where as GM and Hm are 0 as it has zero in its observations

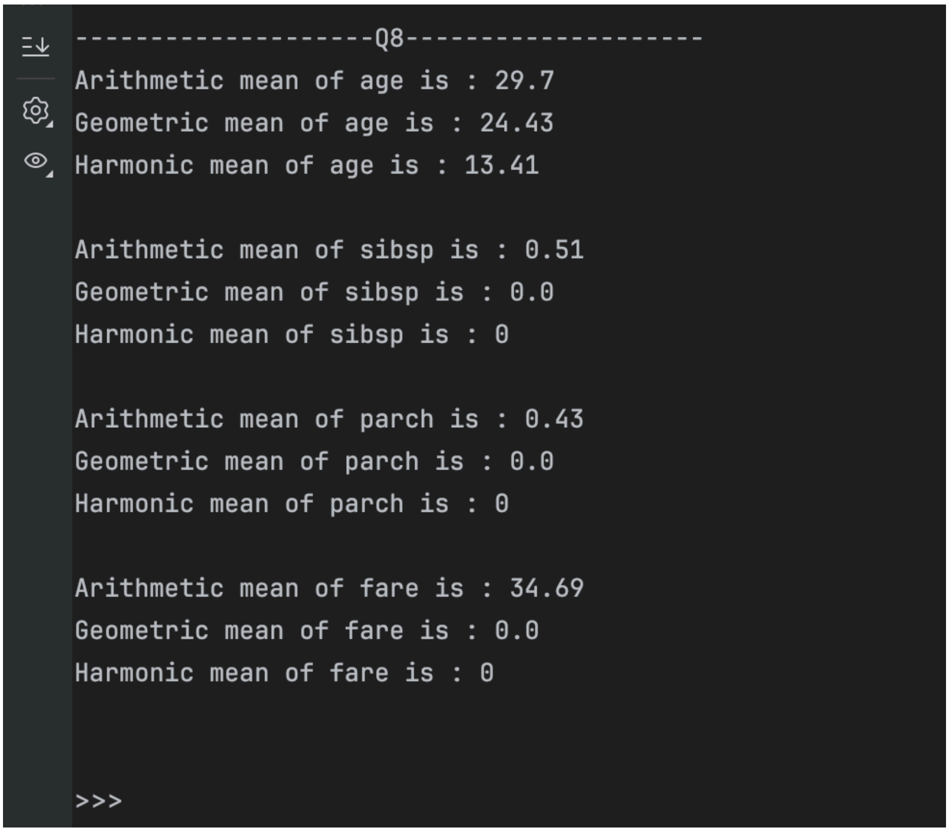
For fare, AM=34.69 where as GM and Hm are 0 as it has zero in its observations

Code:



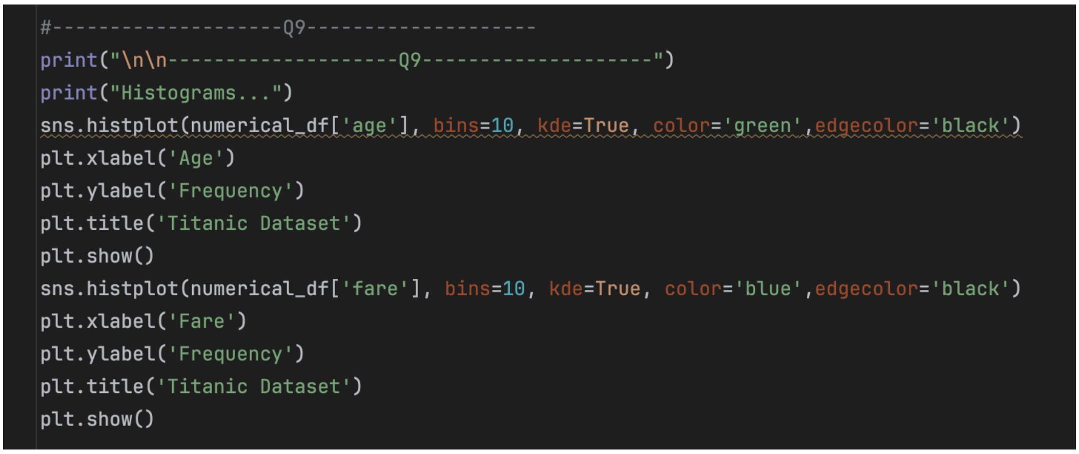
```
#%%--------------------Q8--------------------
from scipy.stats import gmean
from scipy.stats import hmean
print("\n\n--------------------Q8--------------------")
numerical_df=numerical_df.dropna()
def arithmetic_mean(data):  2 usages  new *
    total_sum = sum(data)
    count = len(data)
    return round(total_sum / count, 2)

def geometric_mean(data):  1 usage  new *
    product = 1
    count = len(data)
    for num in data:
        product *= num
    return round(product**(1/count), 2)

def harmonic_mean(data):  1 usage  new *
    if 0 in data or len(data) == 0:
        return 0
    count = len(data)
    reciprocal_sum = sum(1 / num for num in data)
    return round(count / reciprocal_sum, 2)


numerical_df = numerical_df.dropna()
list_cols=['sibsp','parch','fare']
print(f"Arithmetic mean of age is :",arithmetic_mean(data=numerical_df['age']))
print(f"Geometric mean of age is :",round(gmean(numerical_df['age']), 2))
print(f"Harmonic mean of age is :", round(hmean(numerical_df['age']), 2))
print()
for i in list_cols:
    print(f"Arithmetic mean of {i} is :",arithmetic_mean(data=numerical_df[i]))
    print(f"Geometric mean of {i} is :", geometric_mean(data=numerical_df[i]))
    print(f"Harmonic mean of {i} is :", harmonic_mean(data=numerical_df[i]))
    print()
```

Output:



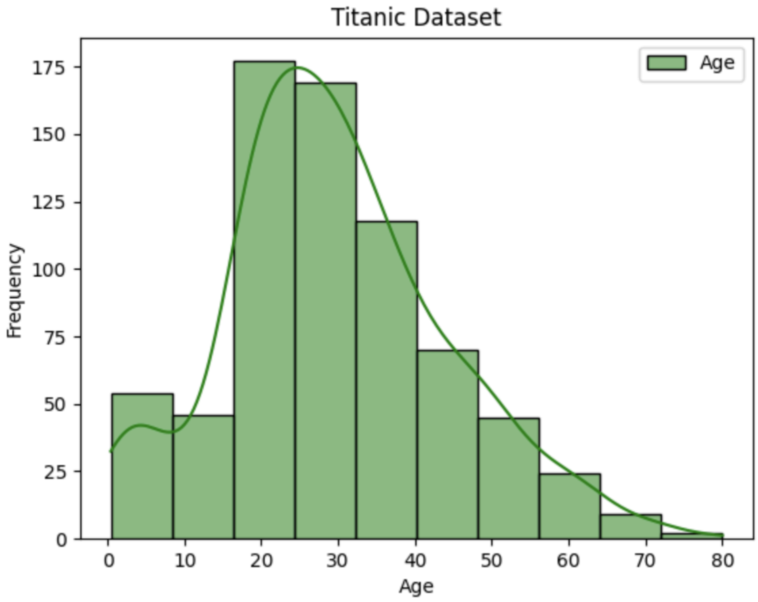
```
--------------------Q8--------------------
Arithmetic mean of age is : 29.7
Geometric mean of age is : 24.43
Harmonic mean of age is : 13.41

Arithmetic mean of sibsp is : 0.51
Geometric mean of sibsp is : 0.0
Harmonic mean of sibsp is : 0

Arithmetic mean of parch is : 0.43
Geometric mean of parch is : 0.0
Harmonic mean of parch is : 0

Arithmetic mean of fare is : 34.69
Geometric mean of fare is : 0.0
Harmonic mean of fare is : 0



>>>
```

Q9.
Code:

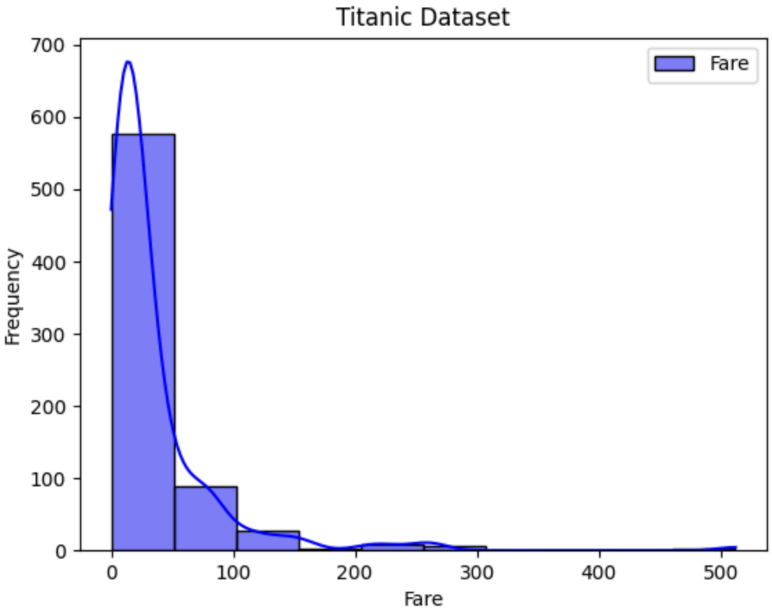


```
#-------------------Q9-------------------
print("\n\n-------------------Q9-------------------")
print("Histograms...")
sns.histplot(numerical_df['age'], bins=10, kde=True, color='green',edgecolor='black')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Titanic Dataset')
plt.show()
sns.histplot(numerical_df['fare'], bins=10, kde=True, color='blue',edgecolor='black')
plt.xlabel('Fare')
plt.ylabel('Frequency')
plt.title('Titanic Dataset')
plt.show()
```

Output:

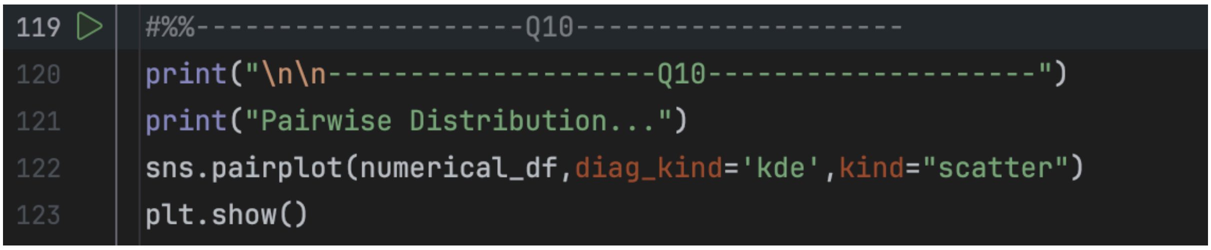


Observations: Majority of the passengers boarded the titanic within age group 18 to 32. There are very few passengers in the age group of 64 to 80. Distribution is right skewed means more passengers are below 40.



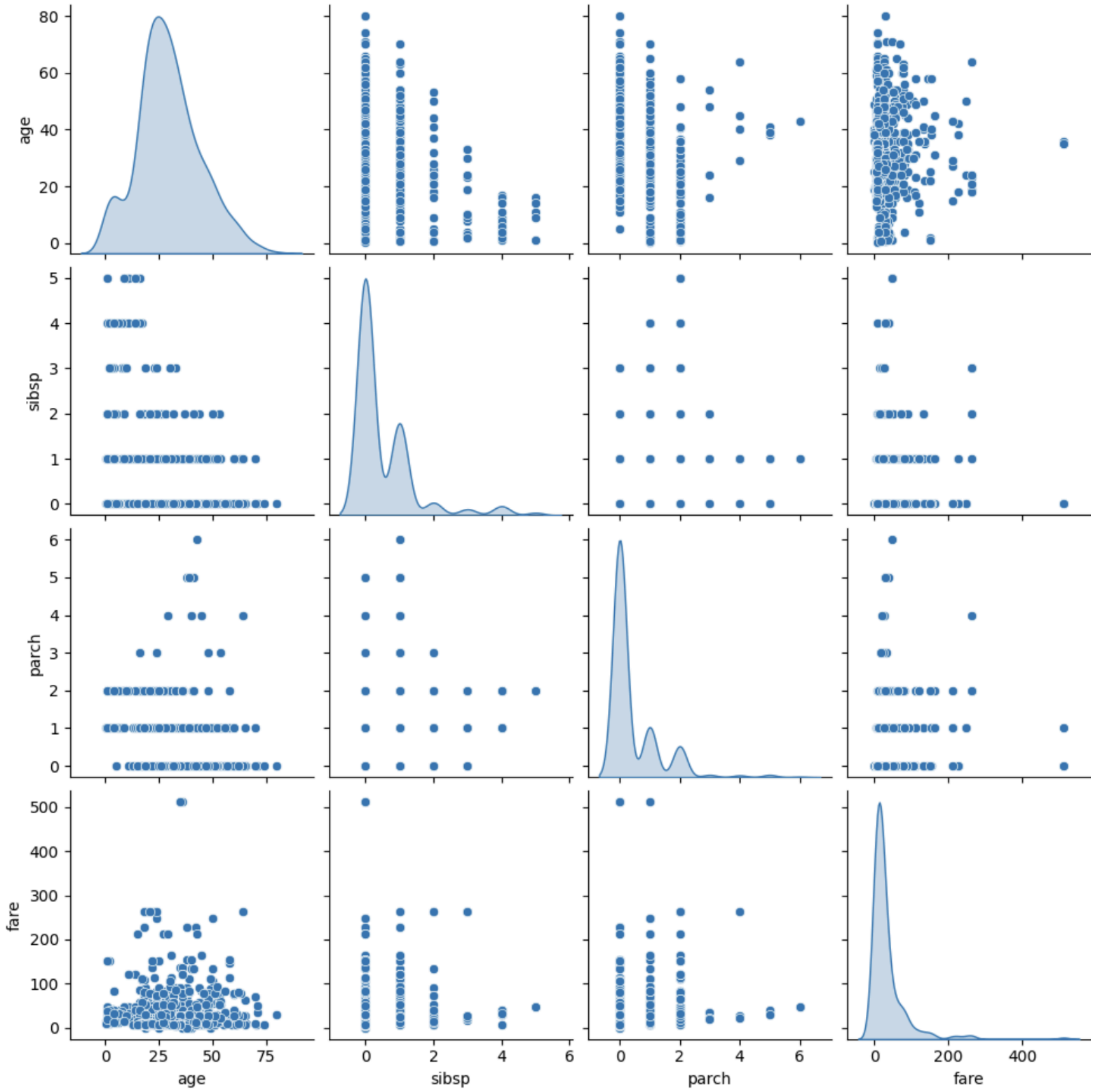
Observations: Majority of passengers paid fares between 0 and 50. Distribution is right-skewed, with a long tail toward higher fares.Very few passengers paid fares over 200.Outliers exist for fares above 300.

Q10.
Code:



```
#%%-------------------Q10-------------------
print("\n\n-------------------Q10-------------------")
print("Pairwise Distribution...")
sns.pairplot(numerical_df,diag_kind='kde',kind="scatter")
plt.show()
```

Output:



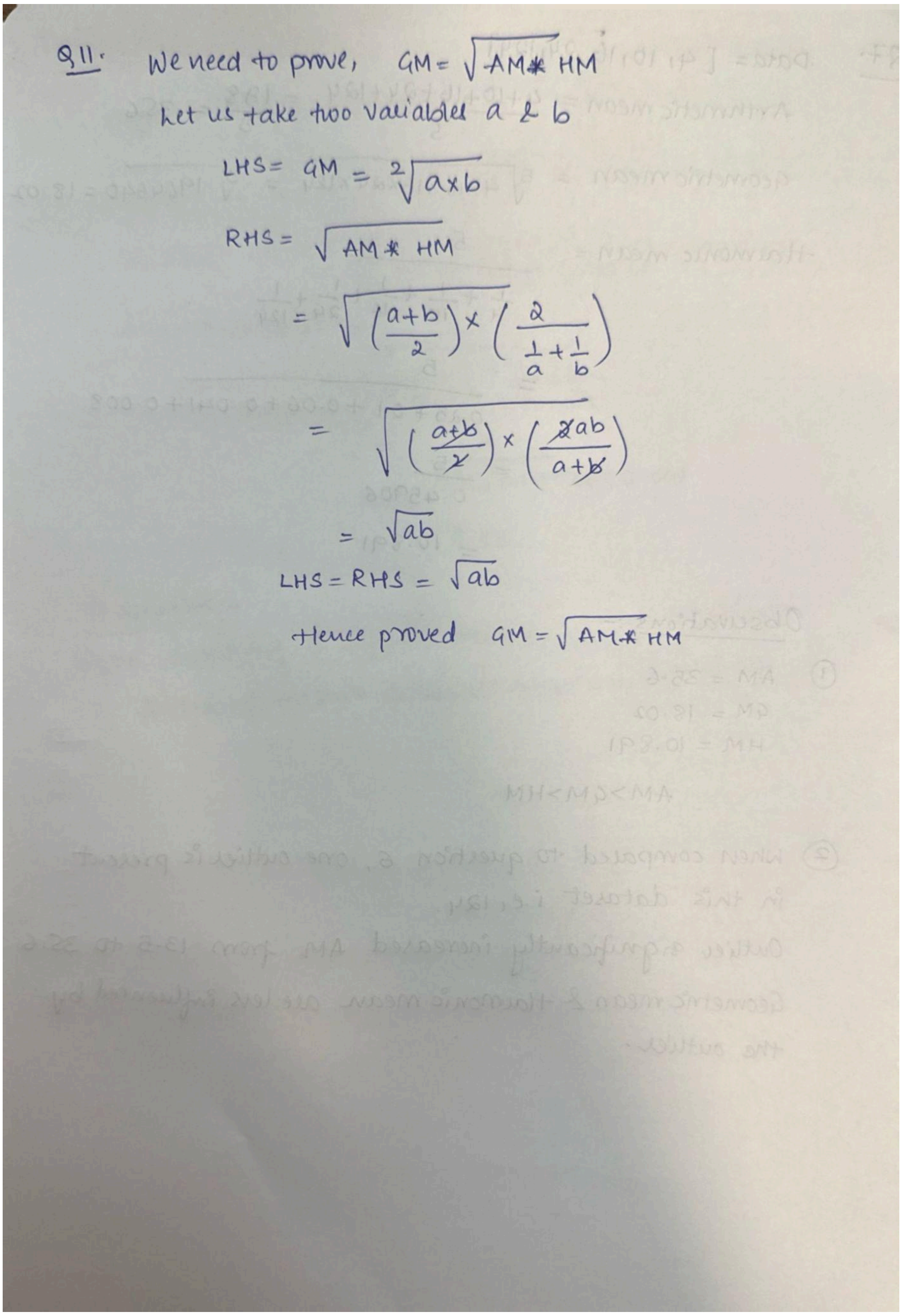
Observations:

1.For the age attribute most of the values falls with the range of 0.42 to 80.

2.sibsp have the min value of 0 and the max value of 5.0. Age got higher range when

sibsp 0 and lower range when sibsp is 4 and 5.
3. Parch have the range between 0 and 6.
4. For all ages most of the fare falls in the range 0 to 100.
5. The most common value of sibsb is 0.
6. The most common value of parch is 0

Q11.



Q11. We need to prove, $GM = \sqrt{AM * HM}$

Let us take two variables a & b

$$LHS = GM = \sqrt[2]{a \times b}$$

$$RHS = \sqrt{AM * HM}$$

$$= \sqrt{\left(\frac{a+b}{2}\right) \times \left(\frac{2}{\frac{1}{a}+\frac{1}{b}}\right)}$$

$$= \sqrt{\left(\frac{\cancel{a+b}}{\cancel{2}}\right) \times \left(\frac{\cancel{2}ab}{\cancel{a+b}}\right)}$$

$$= \sqrt{ab}$$

$$LHS = RHS = \sqrt{ab}$$

Hence proved $GM = \sqrt{AM * HM}$