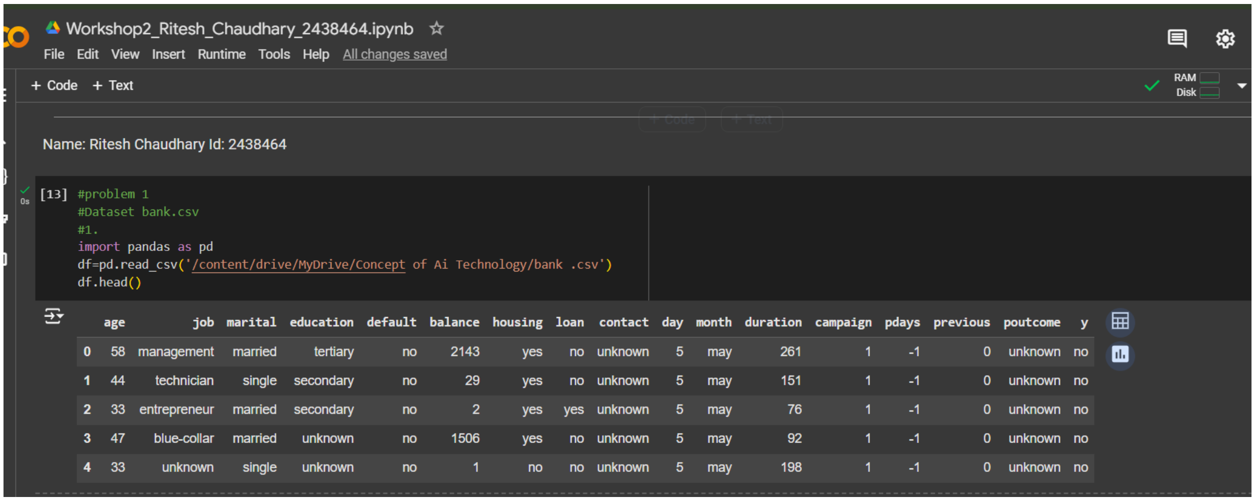
Name: Ritesh Chaudhary

Id: 2438464

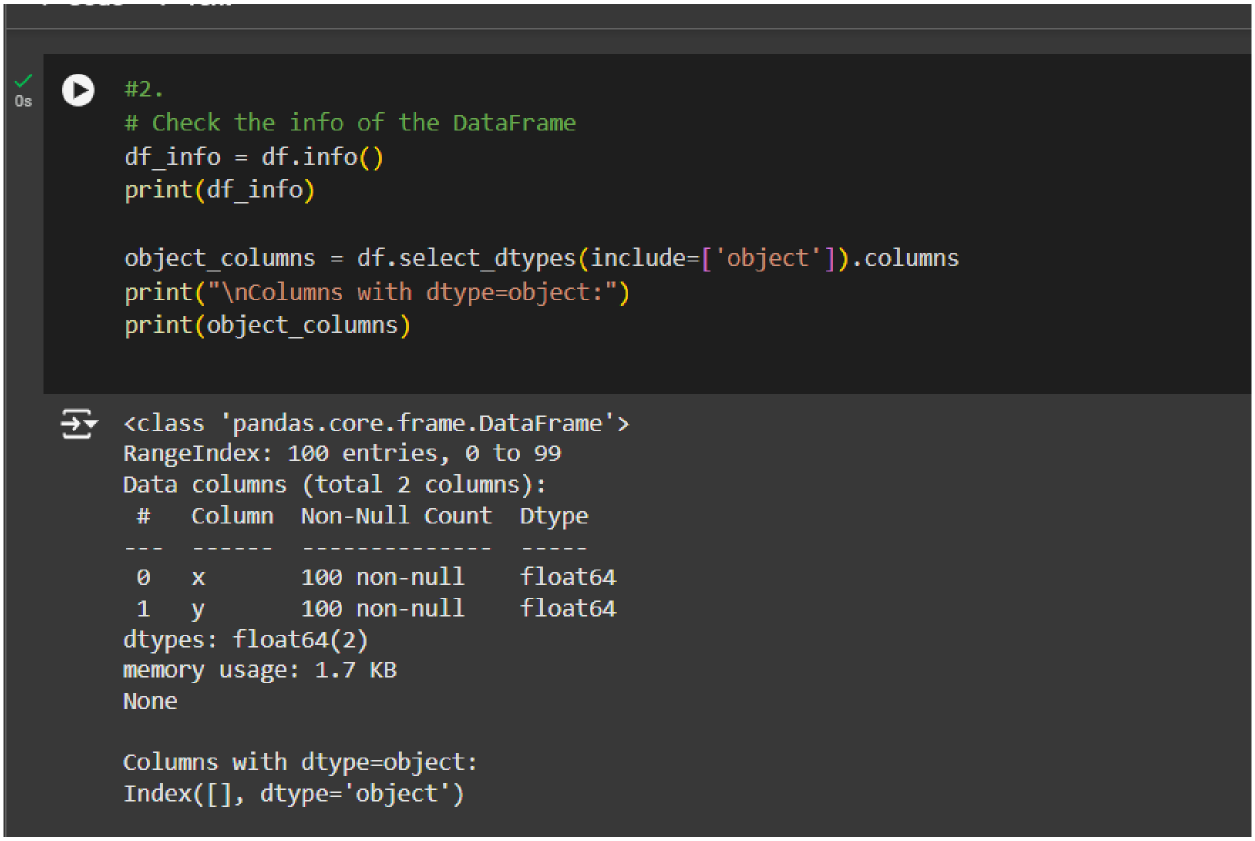
Workshop Week-2


Workshop2_Ritesh_Chaudhary_2438464.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
RAM
Disk

Name: Ritesh Chaudhary Id: 2438464

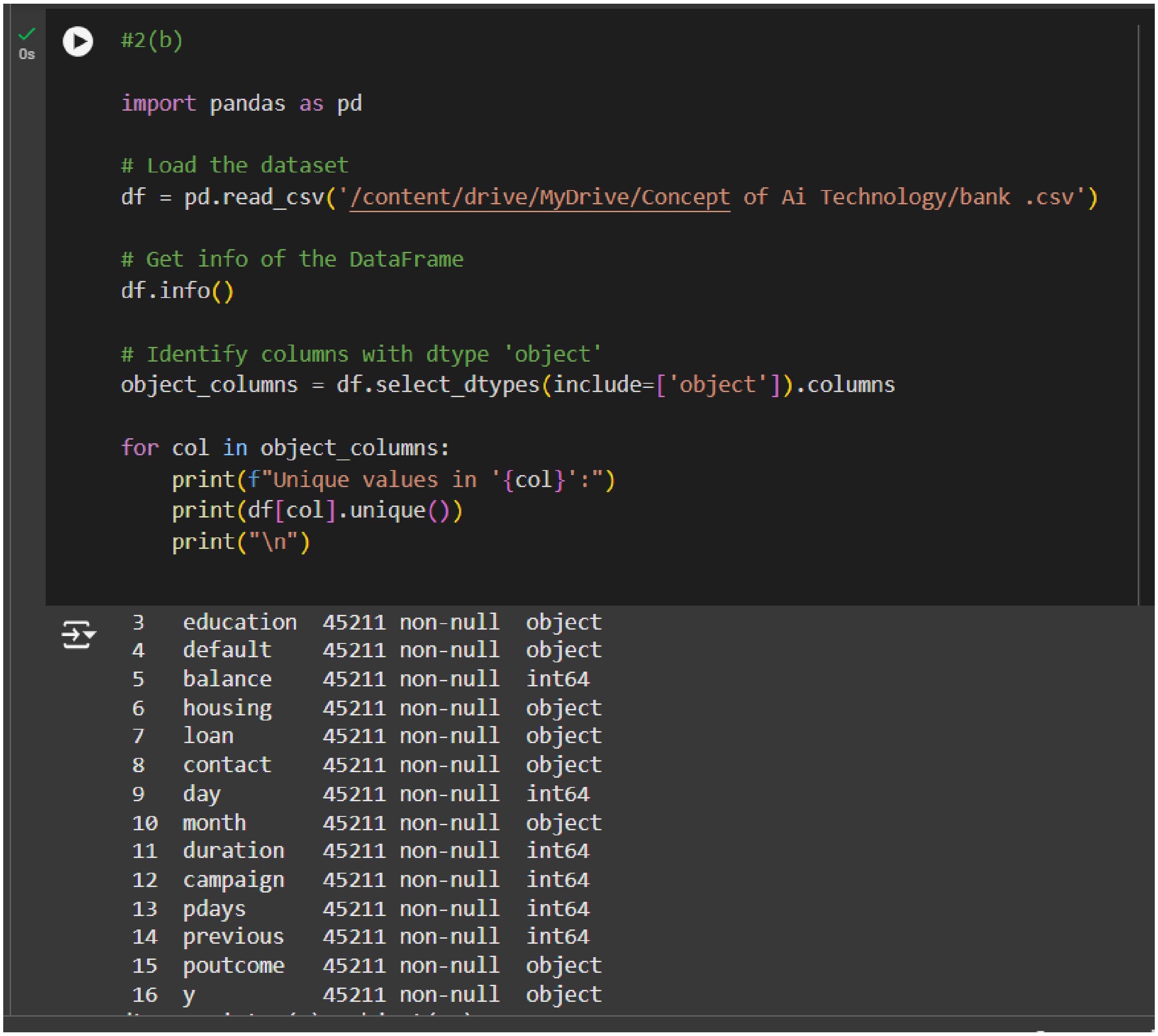
```
[13] #problem 1
     #Dataset bank.csv
     #1.
     import pandas as pd
     df=pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/bank .csv')
     df.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no



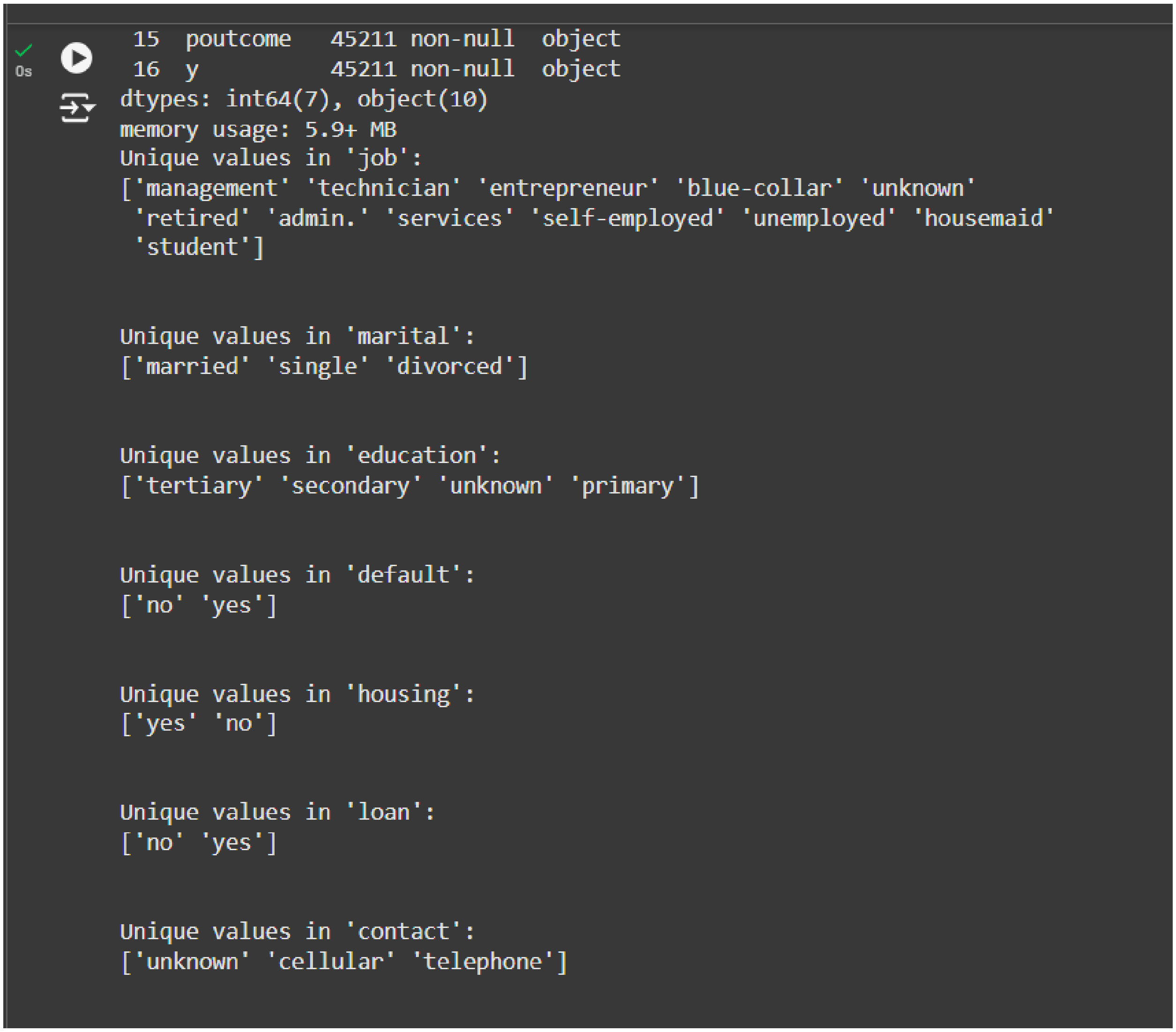
```
#2.
# Check the info of the DataFrame
df_info = df.info()
print(df_info)

object_columns = df.select_dtypes(include=['object']).columns
print("\nColumns with dtype=object:")
print(object_columns)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   x       100 non-null    float64
 1   y       100 non-null    float64
dtypes: float64(2)
memory usage: 1.7 KB
None

Columns with dtype=object:
Index([], dtype='object')
```

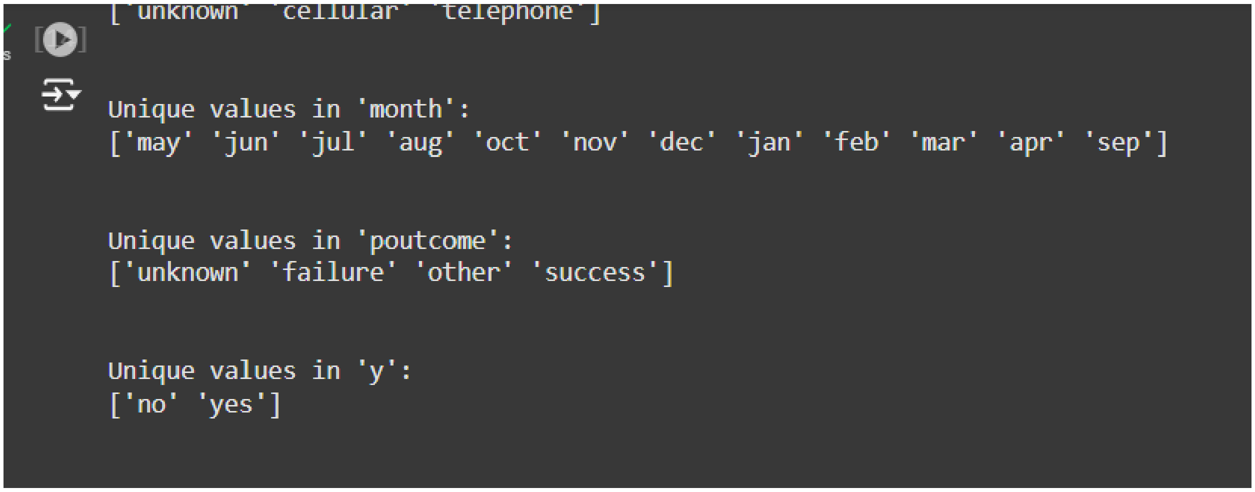


```python
#2(b)

import pandas as pd

# Load the dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/bank .csv')

# Get info of the DataFrame
df.info()

# Identify columns with dtype 'object'
object_columns = df.select_dtypes(include=['object']).columns

for col in object_columns:
    print(f"Unique values in '{col}':")
    print(df[col].unique())
    print("\n")
```

```
3   education  45211 non-null  object
4   default    45211 non-null  object
5   balance    45211 non-null  int64
6   housing    45211 non-null  object
7   loan       45211 non-null  object
8   contact    45211 non-null  object
9   day        45211 non-null  int64
10  month      45211 non-null  object
11  duration   45211 non-null  int64
12  campaign   45211 non-null  int64
13  pdays      45211 non-null  int64
14  previous   45211 non-null  int64
15  poutcome   45211 non-null  object
16  y          45211 non-null  object
```

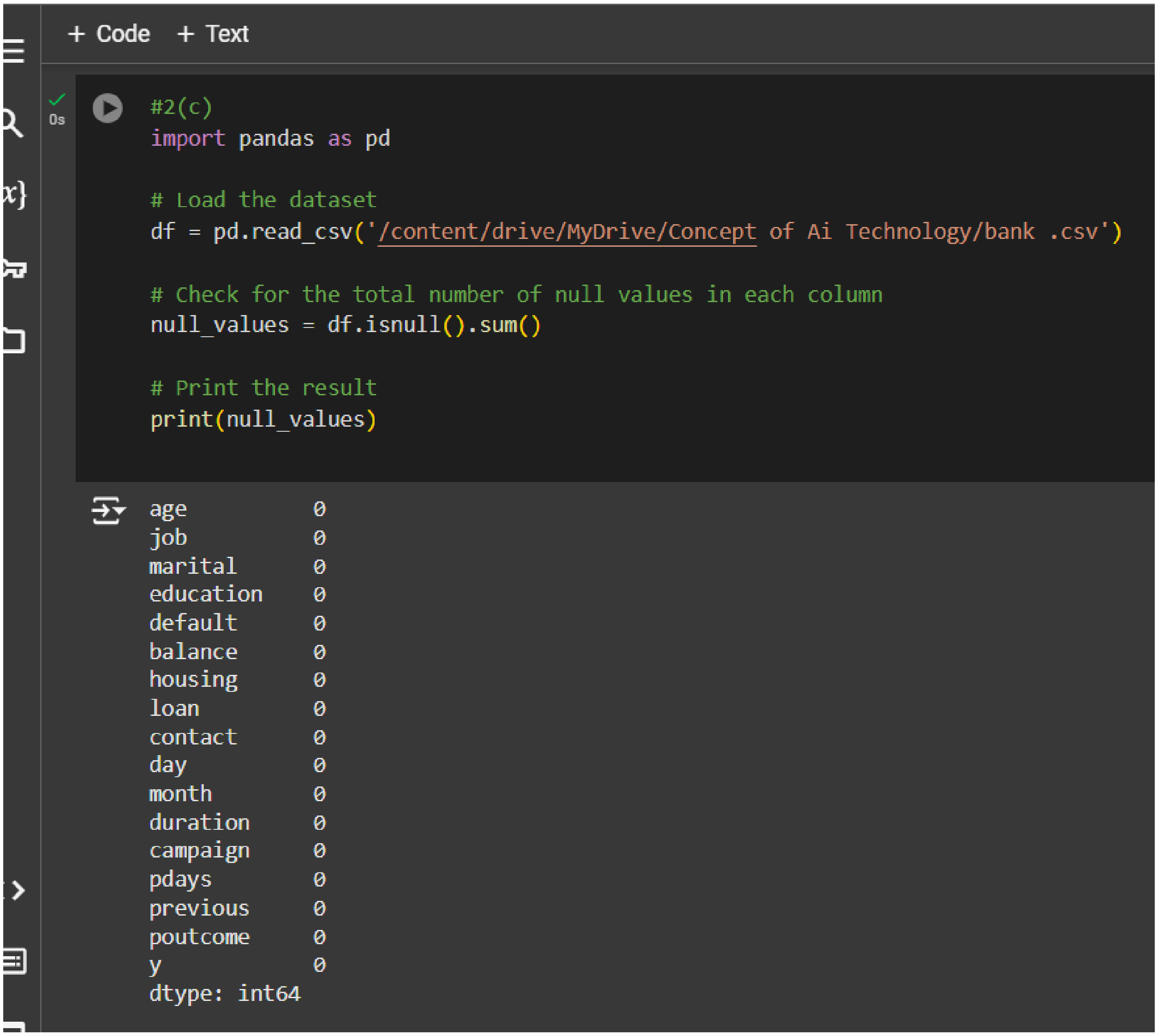


```
 15  poutcome   45211 non-null  object
 16  y          45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
Unique values in 'job':
['management' 'technician' 'entrepreneur' 'blue-collar' 'unknown'
 'retired' 'admin.' 'services' 'self-employed' 'unemployed' 'housemaid'
 'student']


Unique values in 'marital':
['married' 'single' 'divorced']


Unique values in 'education':
['tertiary' 'secondary' 'unknown' 'primary']


Unique values in 'default':
['no' 'yes']


Unique values in 'housing':
['yes' 'no']


Unique values in 'loan':
['no' 'yes']


Unique values in 'contact':
['unknown' 'cellular' 'telephone']
```



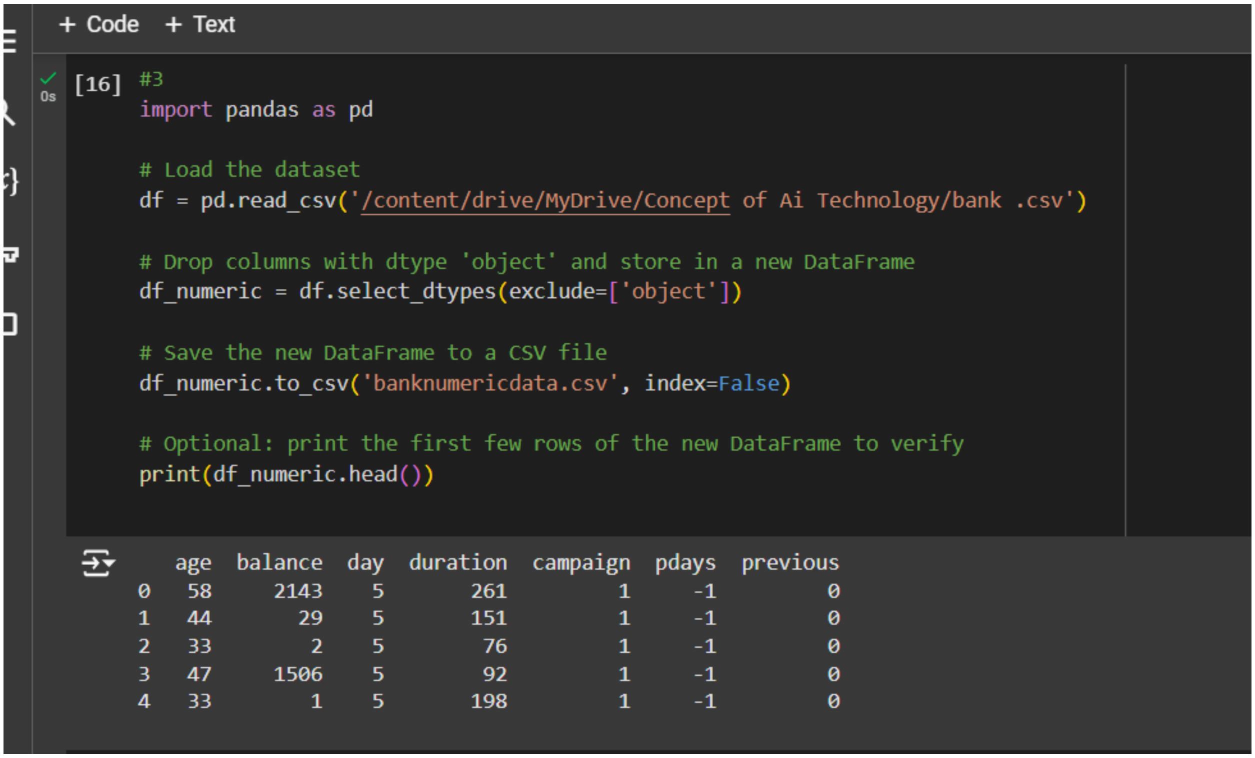
['unknown' 'cellular' 'telephone']

```
Unique values in 'month':
['may' 'jun' 'jul' 'aug' 'oct' 'nov' 'dec' 'jan' 'feb' 'mar' 'apr' 'sep']


Unique values in 'poutcome':
['unknown' 'failure' 'other' 'success']


Unique values in 'y':
['no' 'yes']
```



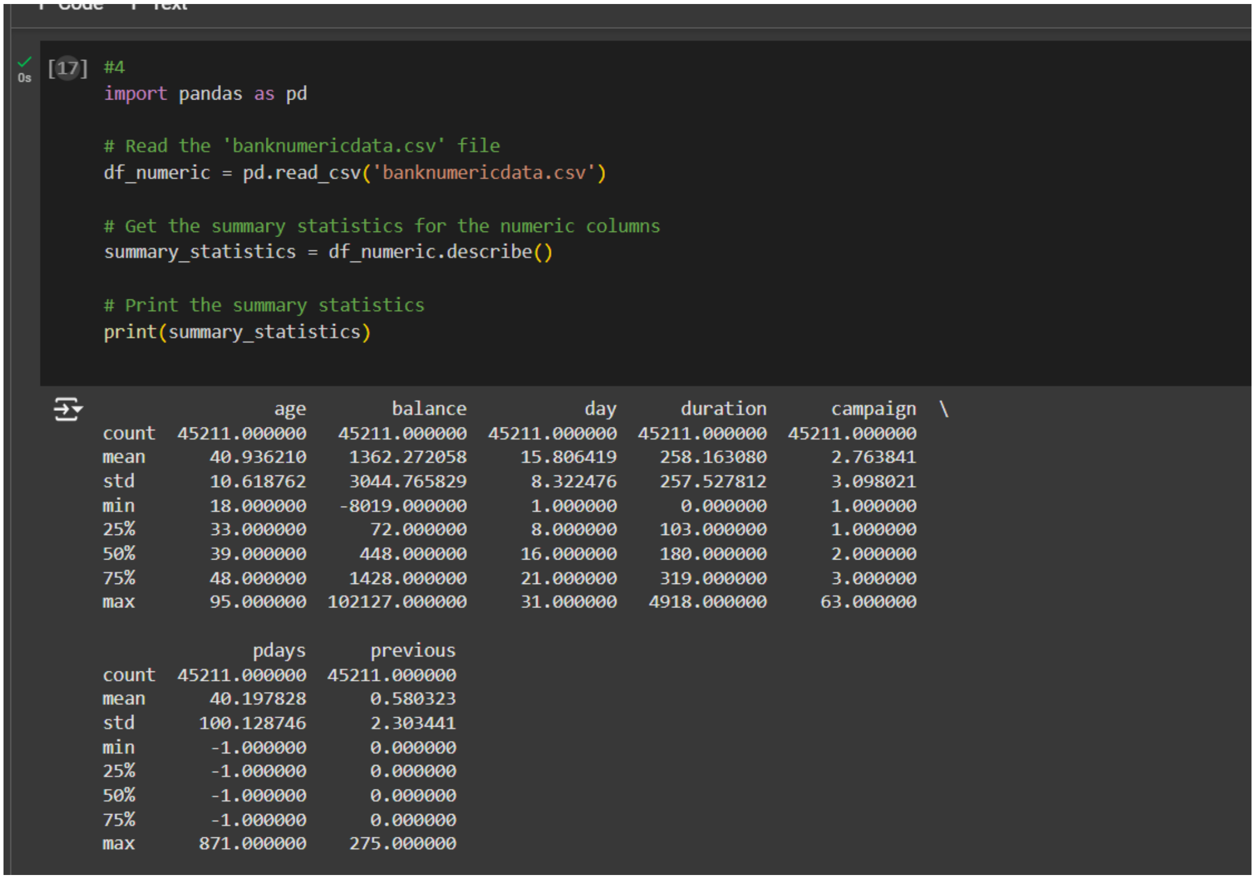
```
#2(c)
import pandas as pd

# Load the dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/bank .csv')

# Check for the total number of null values in each column
null_values = df.isnull().sum()

# Print the result
print(null_values)
```

```
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```



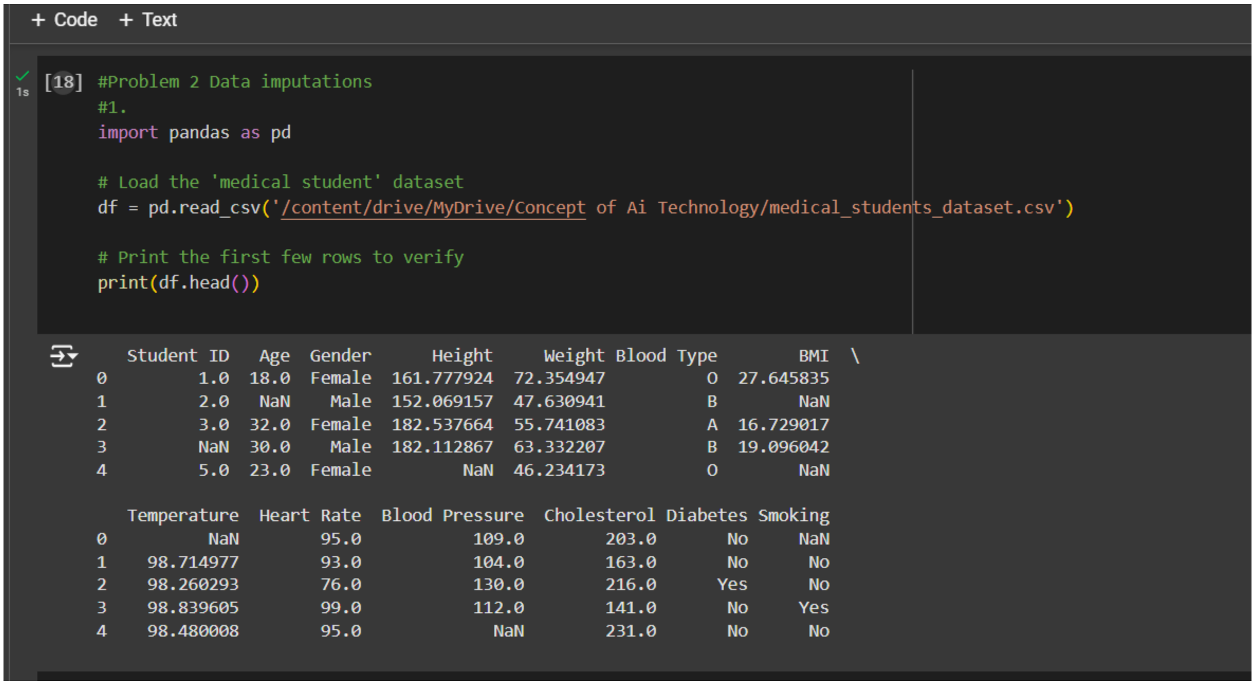
```
#3
import pandas as pd

# Load the dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/bank .csv')

# Drop columns with dtype 'object' and store in a new DataFrame
df_numeric = df.select_dtypes(exclude=['object'])

# Save the new DataFrame to a CSV file
df_numeric.to_csv('banknumericdata.csv', index=False)

# Optional: print the first few rows of the new DataFrame to verify
print(df_numeric.head())
```

```
   age  balance  day  duration  campaign  pdays  previous
0   58     2143    5       261         1     -1         0
1   44       29    5       151         1     -1         0
2   33        2    5        76         1     -1         0
3   47     1506    5        92         1     -1         0
4   33        1    5       198         1     -1         0
```



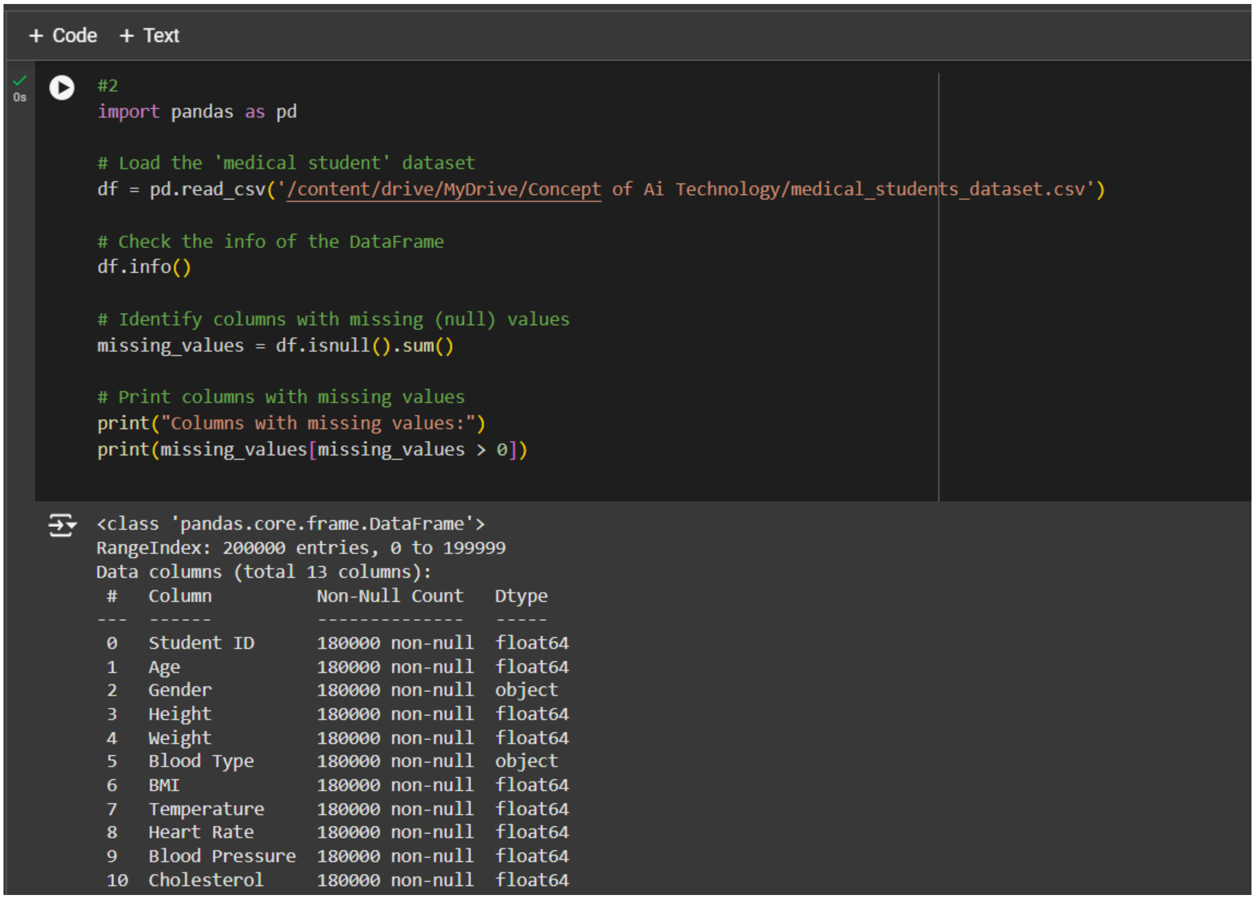
```
#4
import pandas as pd

# Read the 'banknumericdata.csv' file
df_numeric = pd.read_csv('banknumericdata.csv')

# Get the summary statistics for the numeric columns
summary_statistics = df_numeric.describe()

# Print the summary statistics
print(summary_statistics)
```

```
                age        balance           day      duration      campaign  \
count  45211.000000   45211.000000  45211.000000  45211.000000  45211.000000
mean      40.936210    1362.272058     15.806419    258.163080      2.763841
std       10.618762    3044.765829      8.322476    257.527812      3.098021
min       18.000000   -8019.000000      1.000000      0.000000      1.000000
25%       33.000000      72.000000      8.000000    103.000000      1.000000
50%       39.000000     448.000000     16.000000    180.000000      2.000000
75%       48.000000    1428.000000     21.000000    319.000000      3.000000
max       95.000000  102127.000000     31.000000   4918.000000     63.000000

              pdays      previous
count  45211.000000  45211.000000
mean      40.197828      0.580323
std      100.128746      2.303441
min       -1.000000      0.000000
25%       -1.000000      0.000000
50%       -1.000000      0.000000
75%       -1.000000      0.000000
max      871.000000    275.000000
```



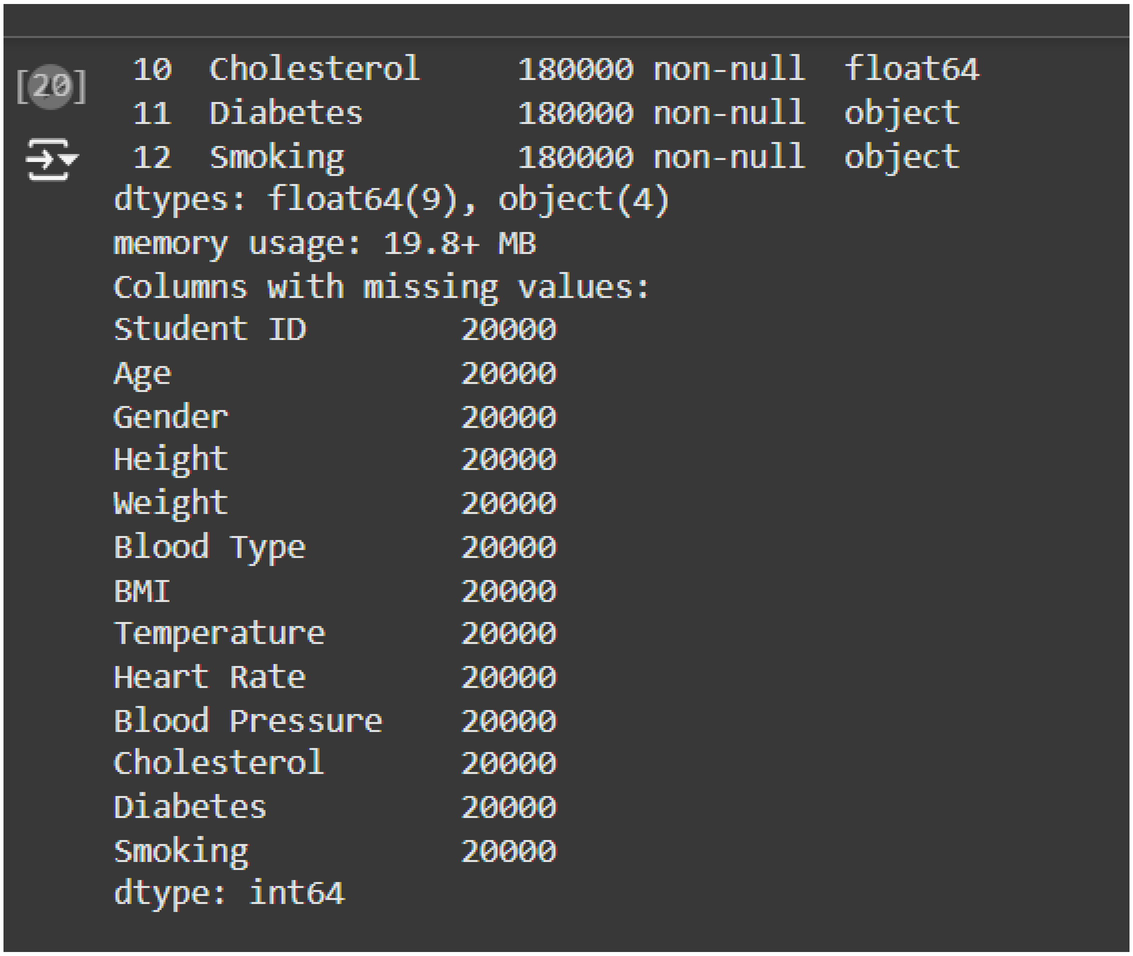
```python
#Problem 2 Data imputations
#1.
import pandas as pd

# Load the 'medical student' dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/medical_students_dataset.csv')

# Print the first few rows to verify
print(df.head())
```

```
   Student ID   Age  Gender      Height     Weight Blood Type        BMI  \
0         1.0  18.0  Female  161.777924  72.354947          O  27.645835
1         2.0   NaN    Male  152.069157  47.630941          B        NaN
2         3.0  32.0  Female  182.537664  55.741083          A  16.729017
3         NaN  30.0    Male  182.112867  63.332207          B  19.096042
4         5.0  23.0  Female         NaN  46.234173          O        NaN

   Temperature  Heart Rate  Blood Pressure  Cholesterol Diabetes Smoking
0          NaN        95.0           109.0        203.0       No     NaN
1    98.714977        93.0           104.0        163.0       No      No
2    98.260293        76.0           130.0        216.0      Yes      No
3    98.839605        99.0           112.0        141.0       No     Yes
4    98.480008        95.0             NaN        231.0       No      No
```

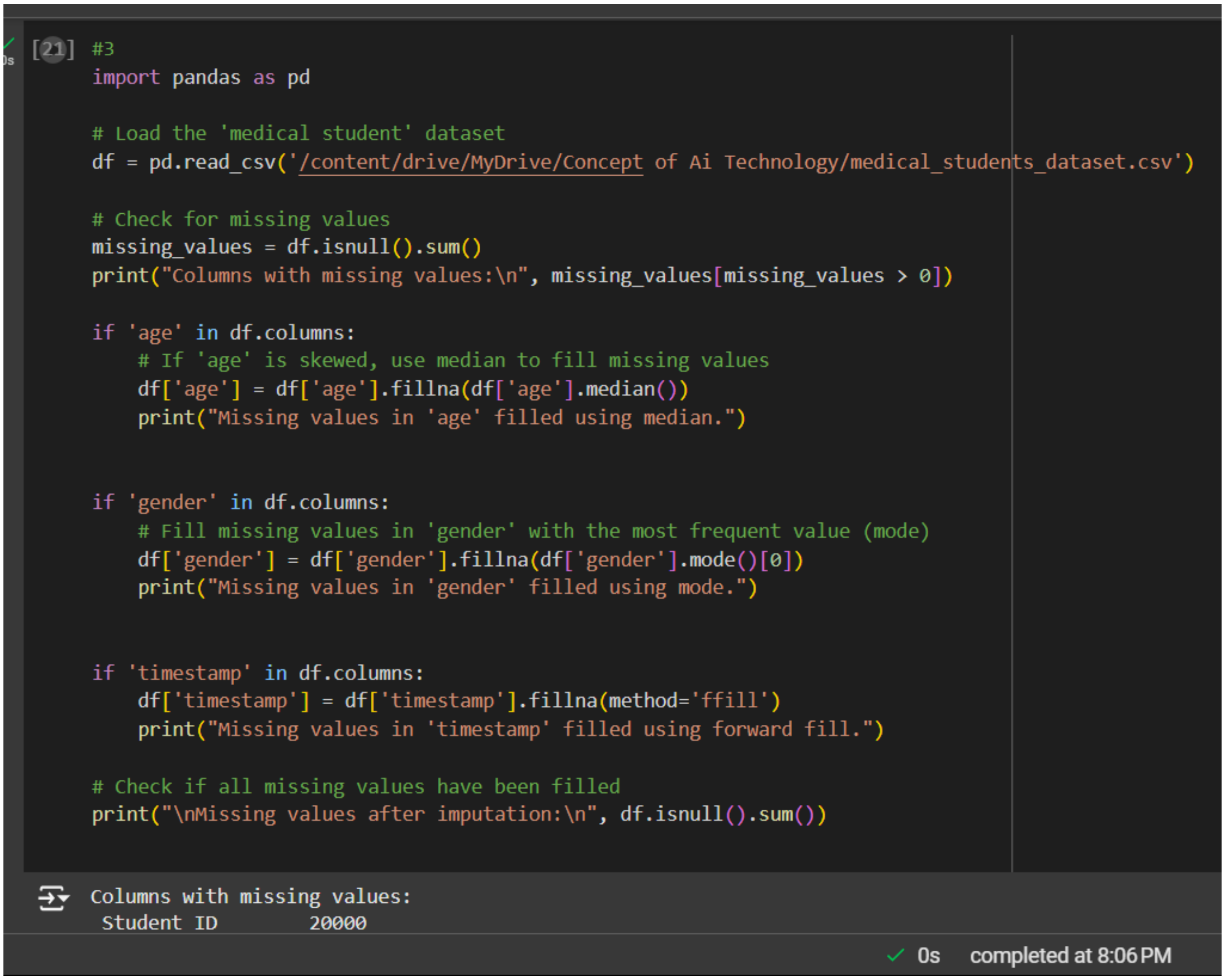


```python
#2
import pandas as pd

# Load the 'medical student' dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/medical_students_dataset.csv')

# Check the info of the DataFrame
df.info()

# Identify columns with missing (null) values
missing_values = df.isnull().sum()

# Print columns with missing values
print("Columns with missing values:")
print(missing_values[missing_values > 0])
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 13 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   Student ID      180000 non-null  float64
 1   Age             180000 non-null  float64
 2   Gender          180000 non-null  object
 3   Height          180000 non-null  float64
 4   Weight          180000 non-null  float64
 5   Blood Type      180000 non-null  object
 6   BMI             180000 non-null  float64
 7   Temperature     180000 non-null  float64
 8   Heart Rate      180000 non-null  float64
 9   Blood Pressure  180000 non-null  float64
 10  Cholesterol     180000 non-null  float64
```

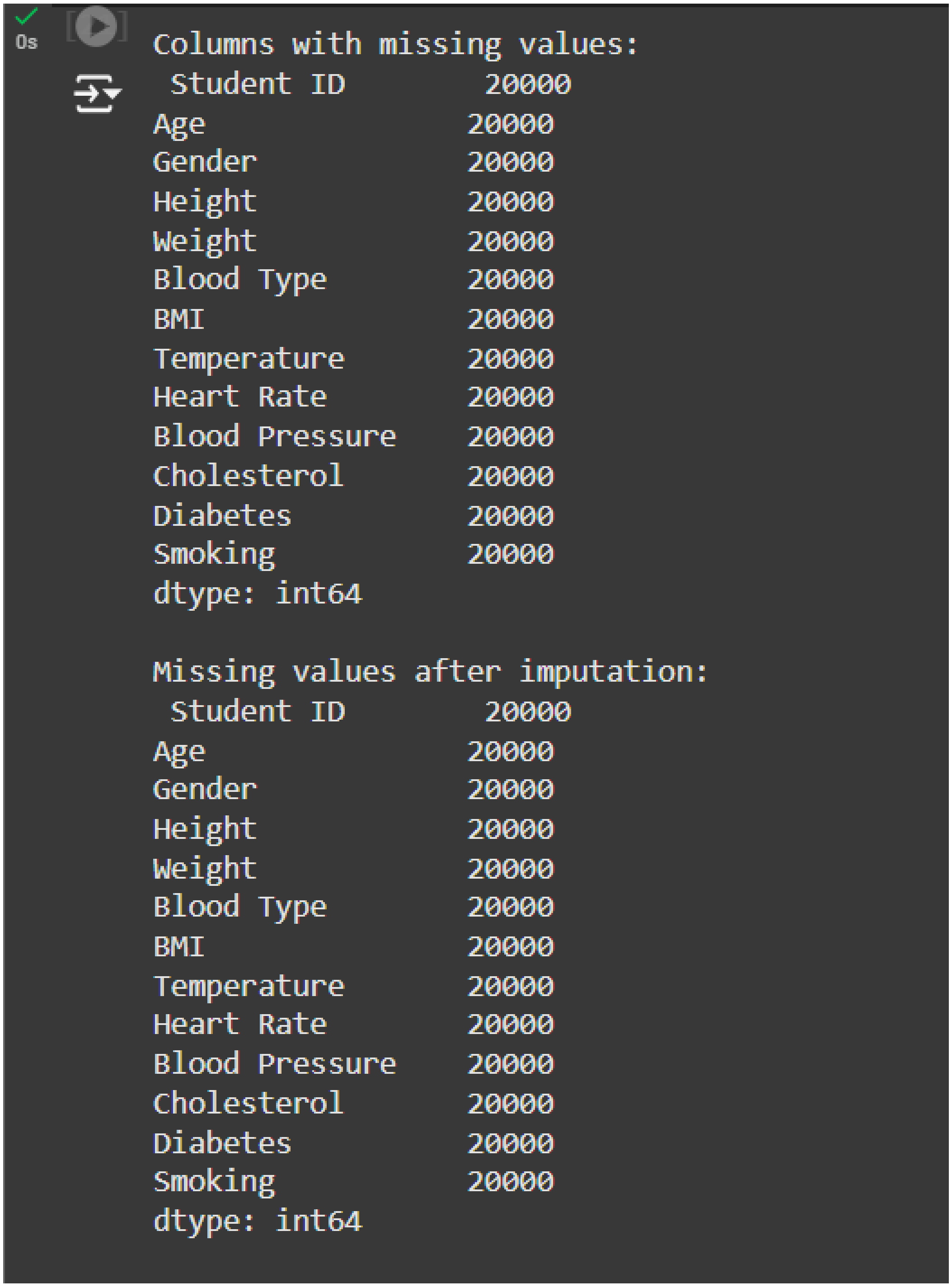


```
 10  Cholesterol     180000 non-null  float64
 11  Diabetes        180000 non-null  object
 12  Smoking         180000 non-null  object
dtypes: float64(9), object(4)
memory usage: 19.8+ MB
Columns with missing values:
Student ID        20000
Age               20000
Gender            20000
Height            20000
Weight            20000
Blood Type        20000
BMI               20000
Temperature       20000
Heart Rate        20000
Blood Pressure    20000
Cholesterol       20000
Diabetes          20000
Smoking           20000
dtype: int64
```

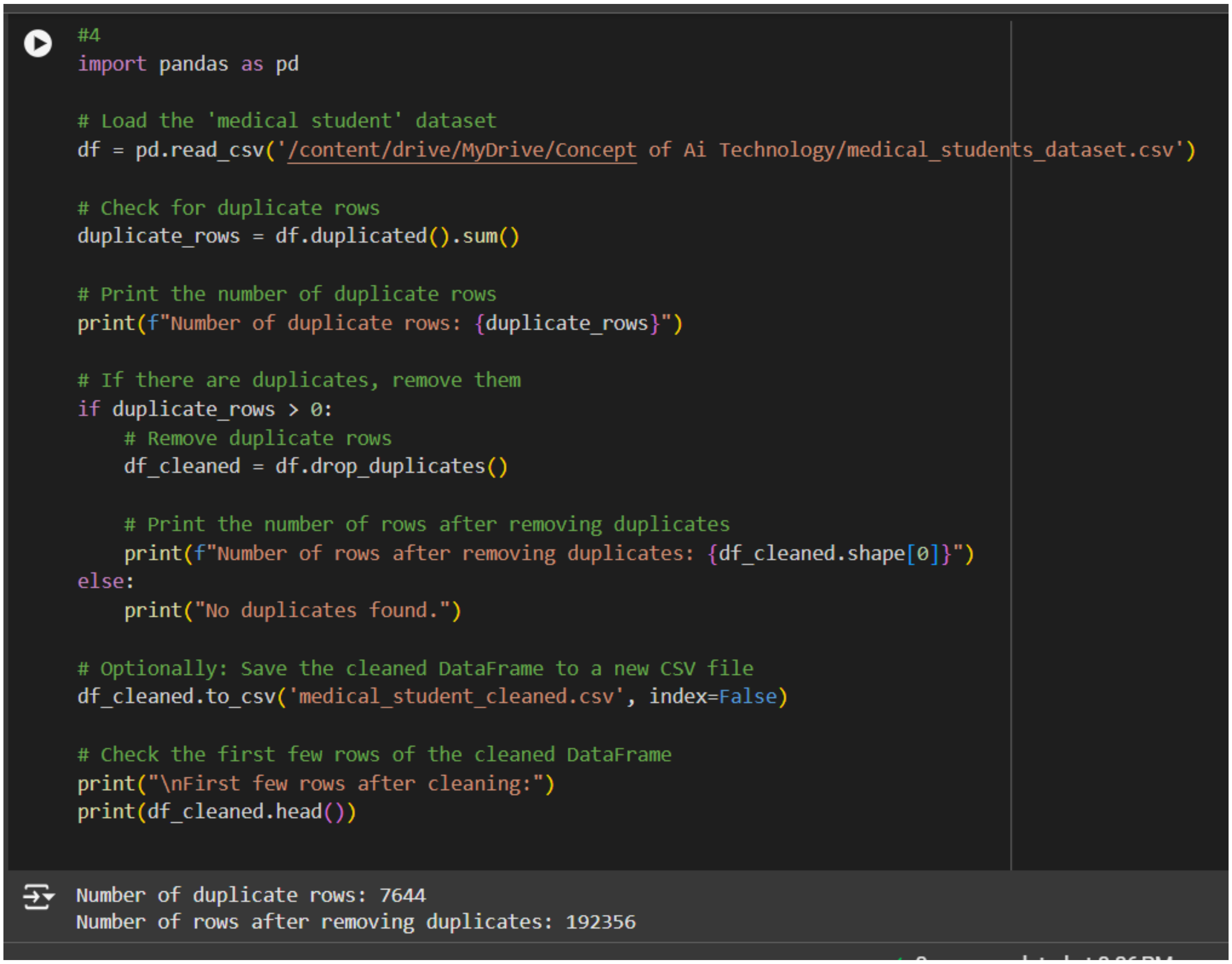


```
#3
import pandas as pd

# Load the 'medical student' dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/medical_students_dataset.csv')

# Check for missing values
missing_values = df.isnull().sum()
print("Columns with missing values:\n", missing_values[missing_values > 0])

if 'age' in df.columns:
    # If 'age' is skewed, use median to fill missing values
    df['age'] = df['age'].fillna(df['age'].median())
    print("Missing values in 'age' filled using median.")


if 'gender' in df.columns:
    # Fill missing values in 'gender' with the most frequent value (mode)
    df['gender'] = df['gender'].fillna(df['gender'].mode()[0])
    print("Missing values in 'gender' filled using mode.")


if 'timestamp' in df.columns:
    df['timestamp'] = df['timestamp'].fillna(method='ffill')
    print("Missing values in 'timestamp' filled using forward fill.")

# Check if all missing values have been filled
print("\nMissing values after imputation:\n", df.isnull().sum())
```

```
Columns with missing values:
 Student ID        20000
```

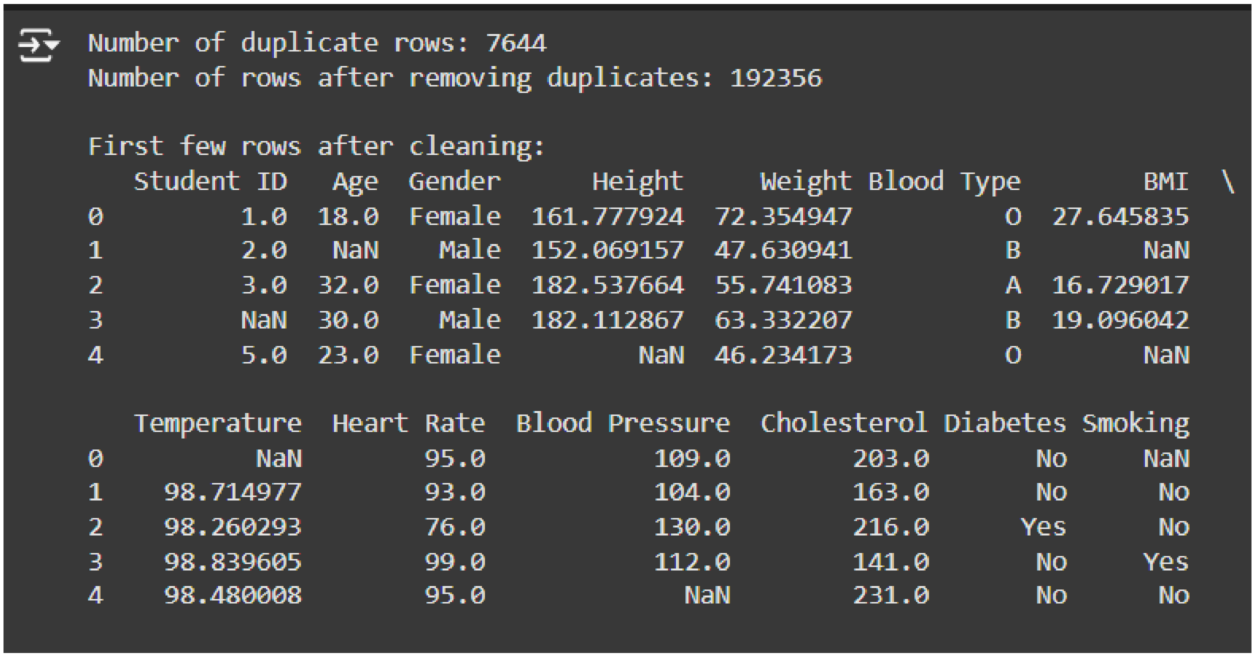


```
Columns with missing values:
 Student ID        20000
Age               20000
Gender            20000
Height            20000
Weight            20000
Blood Type        20000
BMI               20000
Temperature       20000
Heart Rate        20000
Blood Pressure    20000
Cholesterol       20000
Diabetes          20000
Smoking           20000
dtype: int64

Missing values after imputation:
 Student ID        20000
Age               20000
Gender            20000
Height            20000
Weight            20000
Blood Type        20000
BMI               20000
Temperature       20000
Heart Rate        20000
Blood Pressure    20000
Cholesterol       20000
Diabetes          20000
Smoking           20000
dtype: int64
```

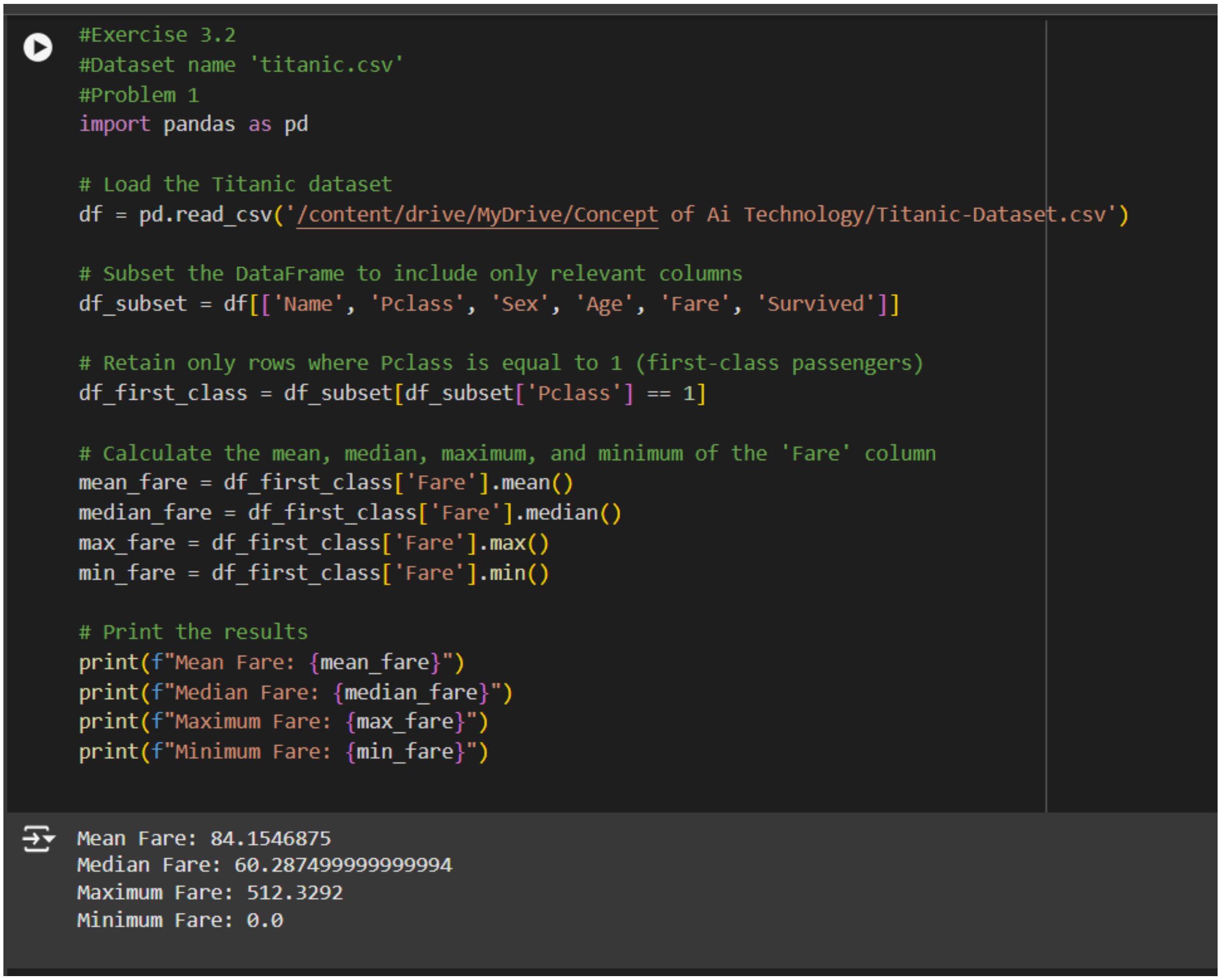


```
#4
import pandas as pd

# Load the 'medical student' dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/medical_students_dataset.csv')

# Check for duplicate rows
duplicate_rows = df.duplicated().sum()

# Print the number of duplicate rows
print(f"Number of duplicate rows: {duplicate_rows}")

# If there are duplicates, remove them
if duplicate_rows > 0:
    # Remove duplicate rows
    df_cleaned = df.drop_duplicates()

    # Print the number of rows after removing duplicates
    print(f"Number of rows after removing duplicates: {df_cleaned.shape[0]}")
else:
    print("No duplicates found.")

# Optionally: Save the cleaned DataFrame to a new CSV file
df_cleaned.to_csv('medical_student_cleaned.csv', index=False)

# Check the first few rows of the cleaned DataFrame
print("\nFirst few rows after cleaning:")
print(df_cleaned.head())
```

```
Number of duplicate rows: 7644
Number of rows after removing duplicates: 192356
```

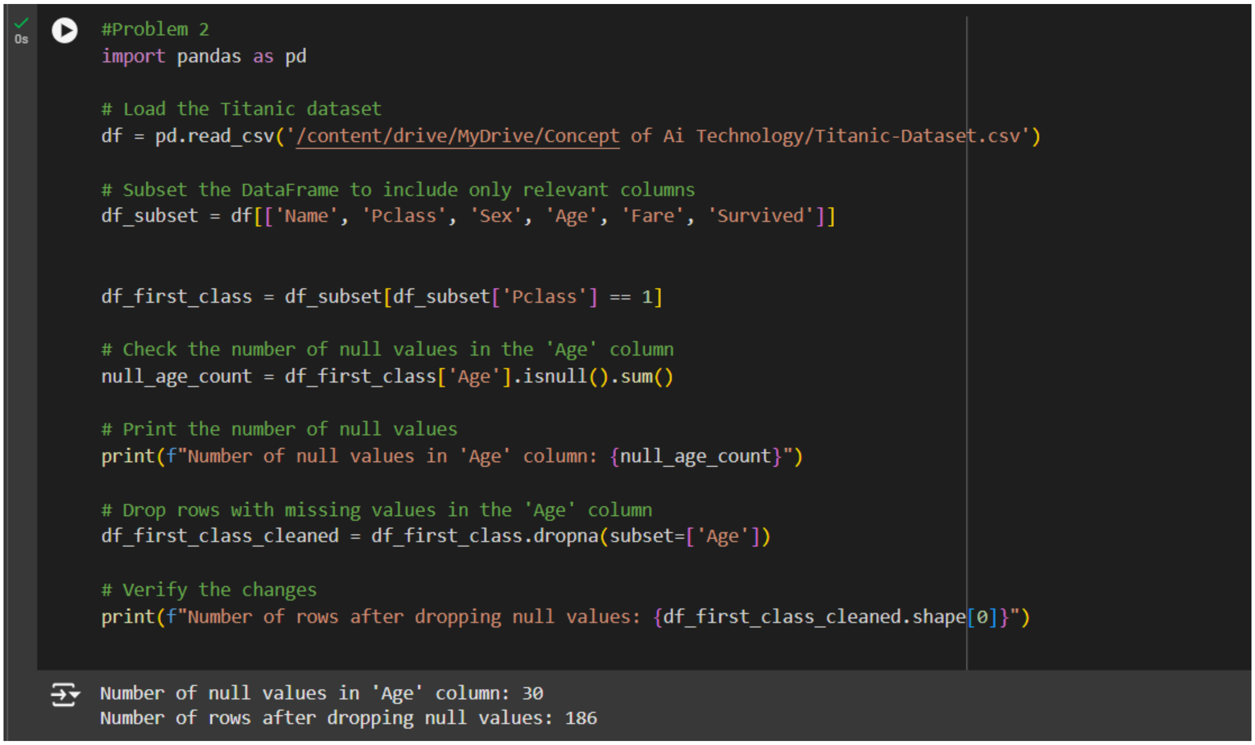


```
Number of duplicate rows: 7644
Number of rows after removing duplicates: 192356

First few rows after cleaning:
   Student ID   Age  Gender      Height     Weight Blood Type        BMI  \
0         1.0  18.0  Female  161.777924  72.354947          O  27.645835
1         2.0   NaN    Male  152.069157  47.630941          B        NaN
2         3.0  32.0  Female  182.537664  55.741083          A  16.729017
3         NaN  30.0    Male  182.112867  63.332207          B  19.096042
4         5.0  23.0  Female         NaN  46.234173          O        NaN

   Temperature  Heart Rate  Blood Pressure  Cholesterol Diabetes Smoking
0          NaN        95.0           109.0        203.0       No     NaN
1    98.714977        93.0           104.0        163.0       No      No
2    98.260293        76.0           130.0        216.0      Yes      No
3    98.839605        99.0           112.0        141.0       No     Yes
4    98.480008        95.0             NaN        231.0       No      No
```



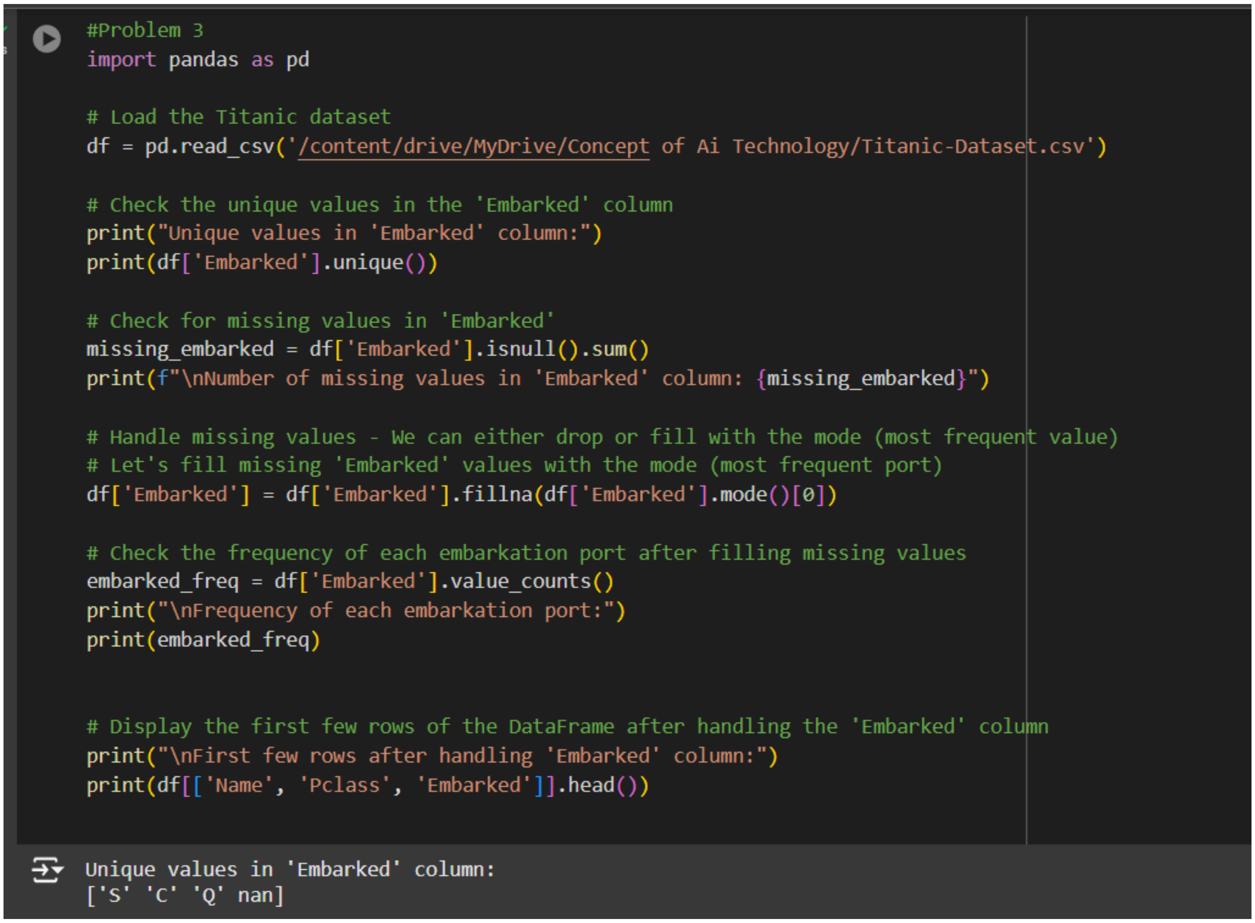
```python
#Exercise 3.2
#Dataset name 'titanic.csv'
#Problem 1
import pandas as pd

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Subset the DataFrame to include only relevant columns
df_subset = df[['Name', 'Pclass', 'Sex', 'Age', 'Fare', 'Survived']]

# Retain only rows where Pclass is equal to 1 (first-class passengers)
df_first_class = df_subset[df_subset['Pclass'] == 1]

# Calculate the mean, median, maximum, and minimum of the 'Fare' column
mean_fare = df_first_class['Fare'].mean()
median_fare = df_first_class['Fare'].median()
max_fare = df_first_class['Fare'].max()
min_fare = df_first_class['Fare'].min()

# Print the results
print(f"Mean Fare: {mean_fare}")
print(f"Median Fare: {median_fare}")
print(f"Maximum Fare: {max_fare}")
print(f"Minimum Fare: {min_fare}")
```

```
Mean Fare: 84.1546875
Median Fare: 60.287499999999994
Maximum Fare: 512.3292
Minimum Fare: 0.0
```



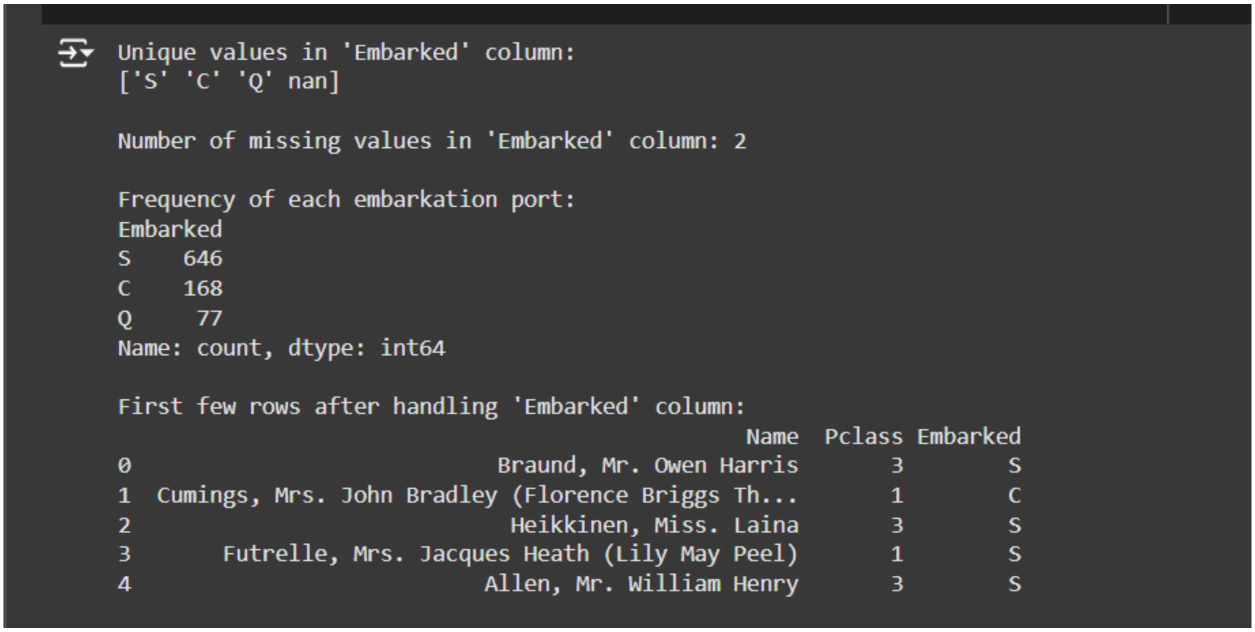
```
#Problem 2
import pandas as pd

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Subset the DataFrame to include only relevant columns
df_subset = df[['Name', 'Pclass', 'Sex', 'Age', 'Fare', 'Survived']]


df_first_class = df_subset[df_subset['Pclass'] == 1]

# Check the number of null values in the 'Age' column
null_age_count = df_first_class['Age'].isnull().sum()

# Print the number of null values
print(f"Number of null values in 'Age' column: {null_age_count}")

# Drop rows with missing values in the 'Age' column
df_first_class_cleaned = df_first_class.dropna(subset=['Age'])

# Verify the changes
print(f"Number of rows after dropping null values: {df_first_class_cleaned.shape[0]}")
```

```
Number of null values in 'Age' column: 30
Number of rows after dropping null values: 186
```

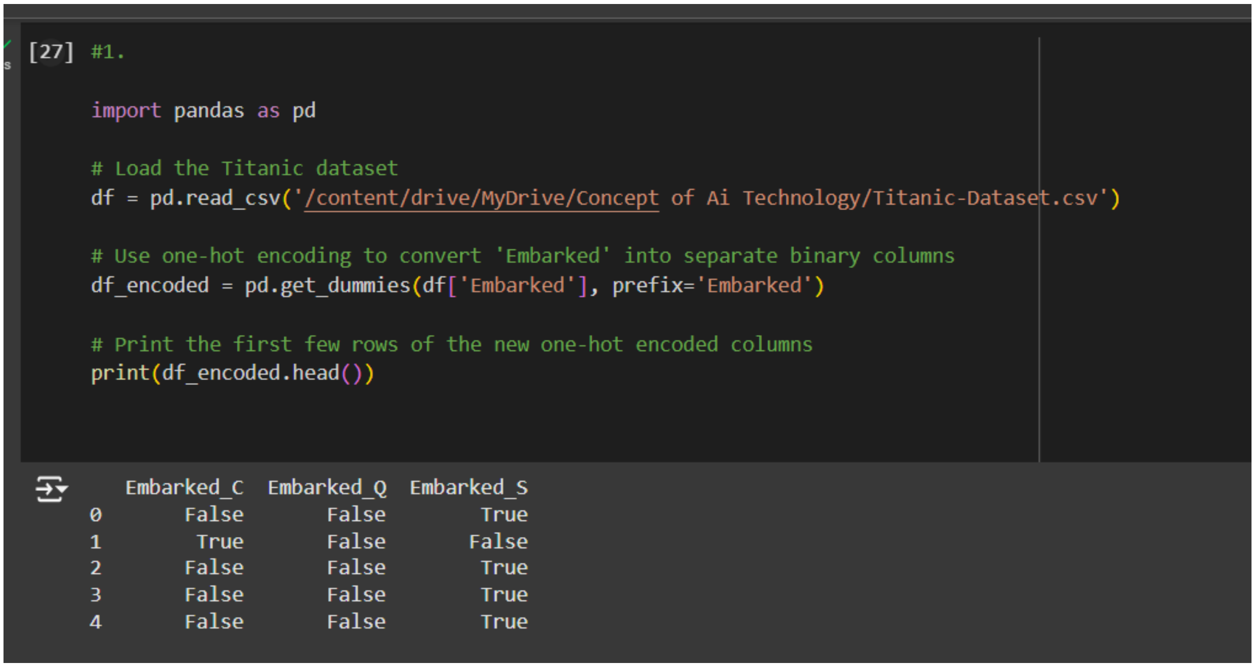


```
#Problem 3
import pandas as pd

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Check the unique values in the 'Embarked' column
print("Unique values in 'Embarked' column:")
print(df['Embarked'].unique())

# Check for missing values in 'Embarked'
missing_embarked = df['Embarked'].isnull().sum()
print(f"\nNumber of missing values in 'Embarked' column: {missing_embarked}")

# Handle missing values - We can either drop or fill with the mode (most frequent value)
# Let's fill missing 'Embarked' values with the mode (most frequent port)
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])

# Check the frequency of each embarkation port after filling missing values
embarked_freq = df['Embarked'].value_counts()
print("\nFrequency of each embarkation port:")
print(embarked_freq)


# Display the first few rows of the DataFrame after handling the 'Embarked' column
print("\nFirst few rows after handling 'Embarked' column:")
print(df[['Name', 'Pclass', 'Embarked']].head())
```

```
Unique values in 'Embarked' column:
['S' 'C' 'Q' nan]
```

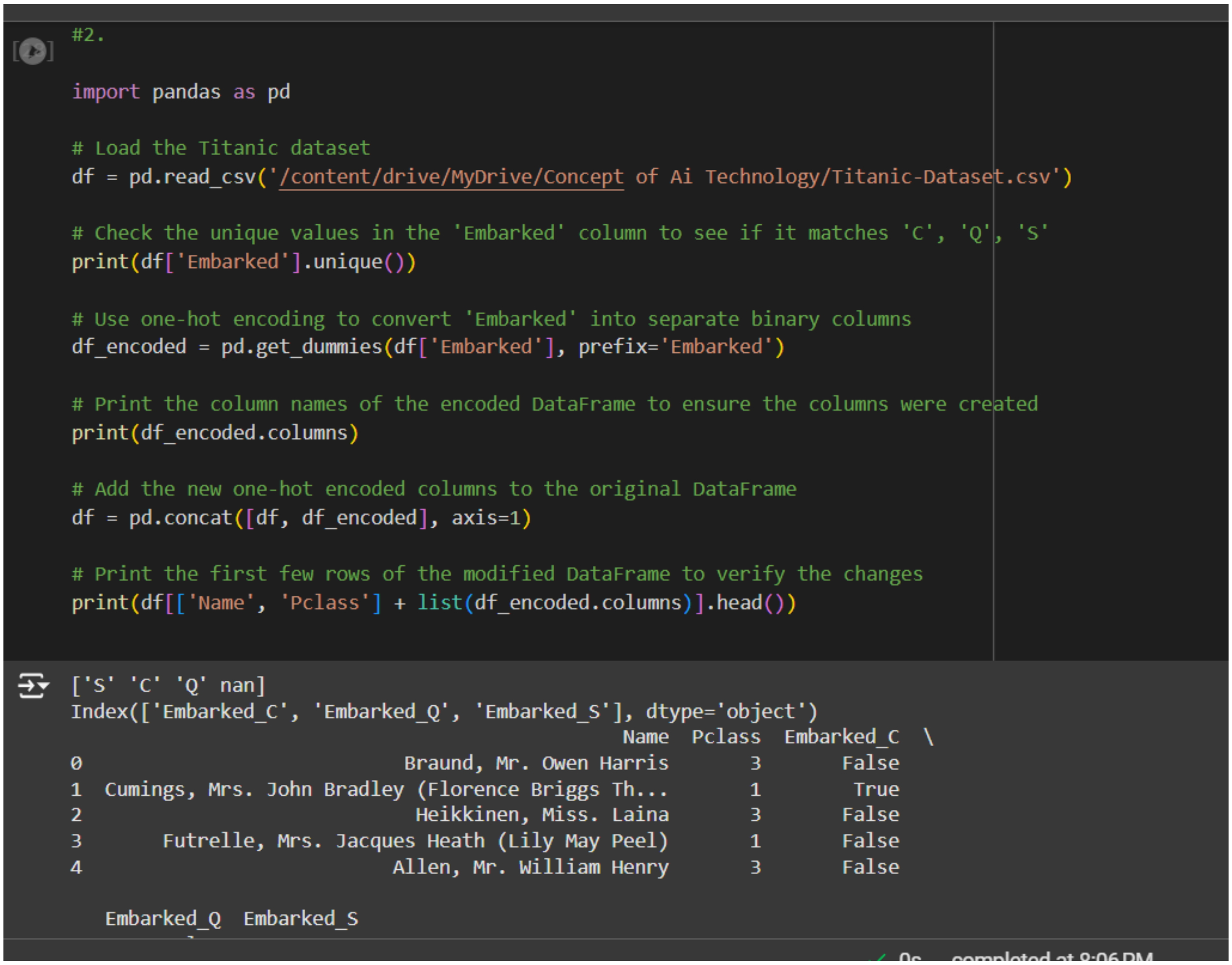


```
Unique values in 'Embarked' column:
['S' 'C' 'Q' nan]

Number of missing values in 'Embarked' column: 2

Frequency of each embarkation port:
Embarked
S    646
C    168
Q     77
Name: count, dtype: int64

First few rows after handling 'Embarked' column:
                                                Name  Pclass Embarked
0                            Braund, Mr. Owen Harris       3        S
1  Cumings, Mrs. John Bradley (Florence Briggs Th...       1        C
2                             Heikkinen, Miss. Laina       3        S
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)       1        S
4                           Allen, Mr. William Henry       3        S
```



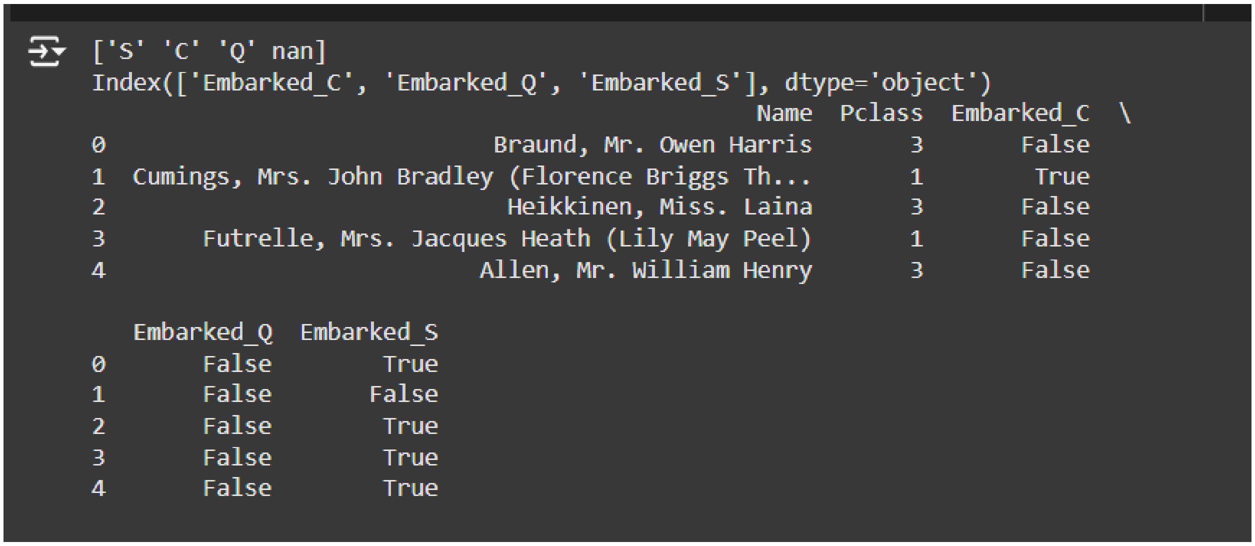
```
[27] #1.

     import pandas as pd

     # Load the Titanic dataset
     df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

     # Use one-hot encoding to convert 'Embarked' into separate binary columns
     df_encoded = pd.get_dummies(df['Embarked'], prefix='Embarked')

     # Print the first few rows of the new one-hot encoded columns
     print(df_encoded.head())
```

```
   Embarked_C  Embarked_Q  Embarked_S
0       False       False        True
1        True       False       False
2       False       False        True
3       False       False        True
4       False       False        True
```

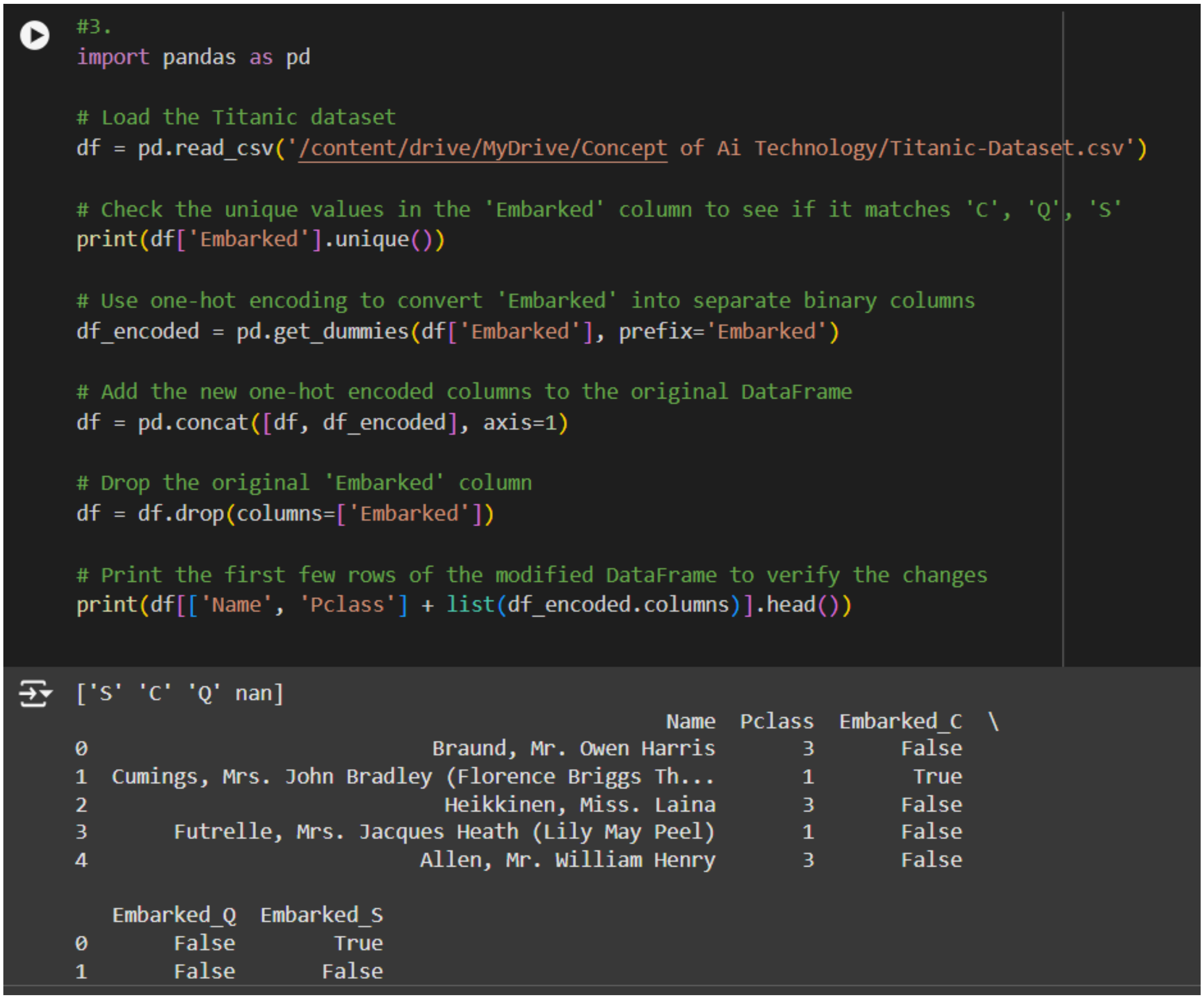


```
#2.

import pandas as pd

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Check the unique values in the 'Embarked' column to see if it matches 'C', 'Q', 'S'
print(df['Embarked'].unique())

# Use one-hot encoding to convert 'Embarked' into separate binary columns
df_encoded = pd.get_dummies(df['Embarked'], prefix='Embarked')

# Print the column names of the encoded DataFrame to ensure the columns were created
print(df_encoded.columns)

# Add the new one-hot encoded columns to the original DataFrame
df = pd.concat([df, df_encoded], axis=1)

# Print the first few rows of the modified DataFrame to verify the changes
print(df[['Name', 'Pclass'] + list(df_encoded.columns)].head())
```

```
['S' 'C' 'Q' nan]
Index(['Embarked_C', 'Embarked_Q', 'Embarked_S'], dtype='object')
                                                Name  Pclass  Embarked_C  \
0                            Braund, Mr. Owen Harris       3       False
1  Cumings, Mrs. John Bradley (Florence Briggs Th...       1        True
2                             Heikkinen, Miss. Laina       3       False
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)       1       False
4                           Allen, Mr. William Henry       3       False

   Embarked_Q  Embarked_S
```

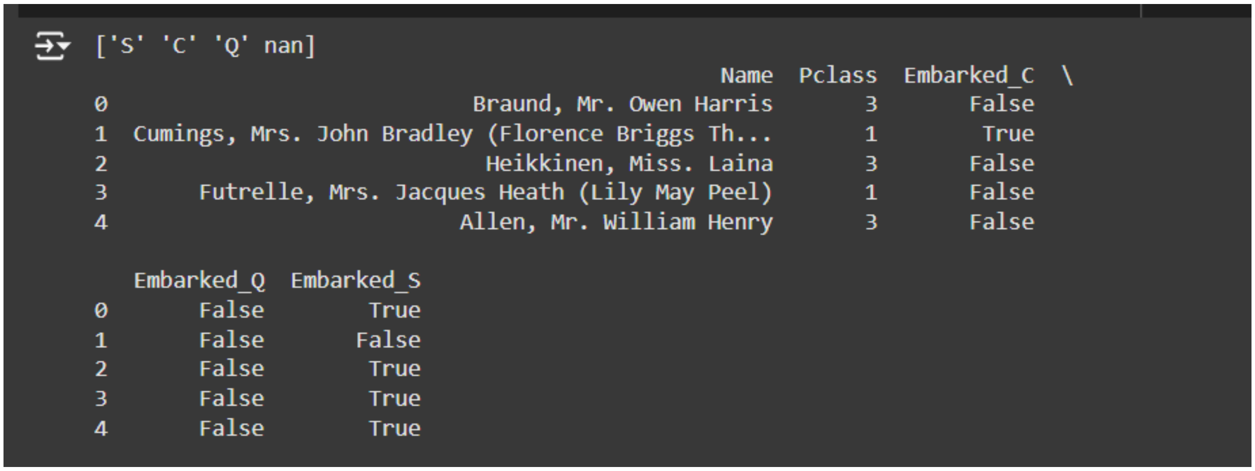


```
['S' 'C' 'Q' nan]
Index(['Embarked_C', 'Embarked_Q', 'Embarked_S'], dtype='object')
                                                Name  Pclass  Embarked_C  \
0                            Braund, Mr. Owen Harris       3       False
1  Cumings, Mrs. John Bradley (Florence Briggs Th...       1        True
2                             Heikkinen, Miss. Laina       3       False
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)       1       False
4                           Allen, Mr. William Henry       3       False

   Embarked_Q  Embarked_S
0       False        True
1       False       False
2       False        True
3       False        True
4       False        True
```

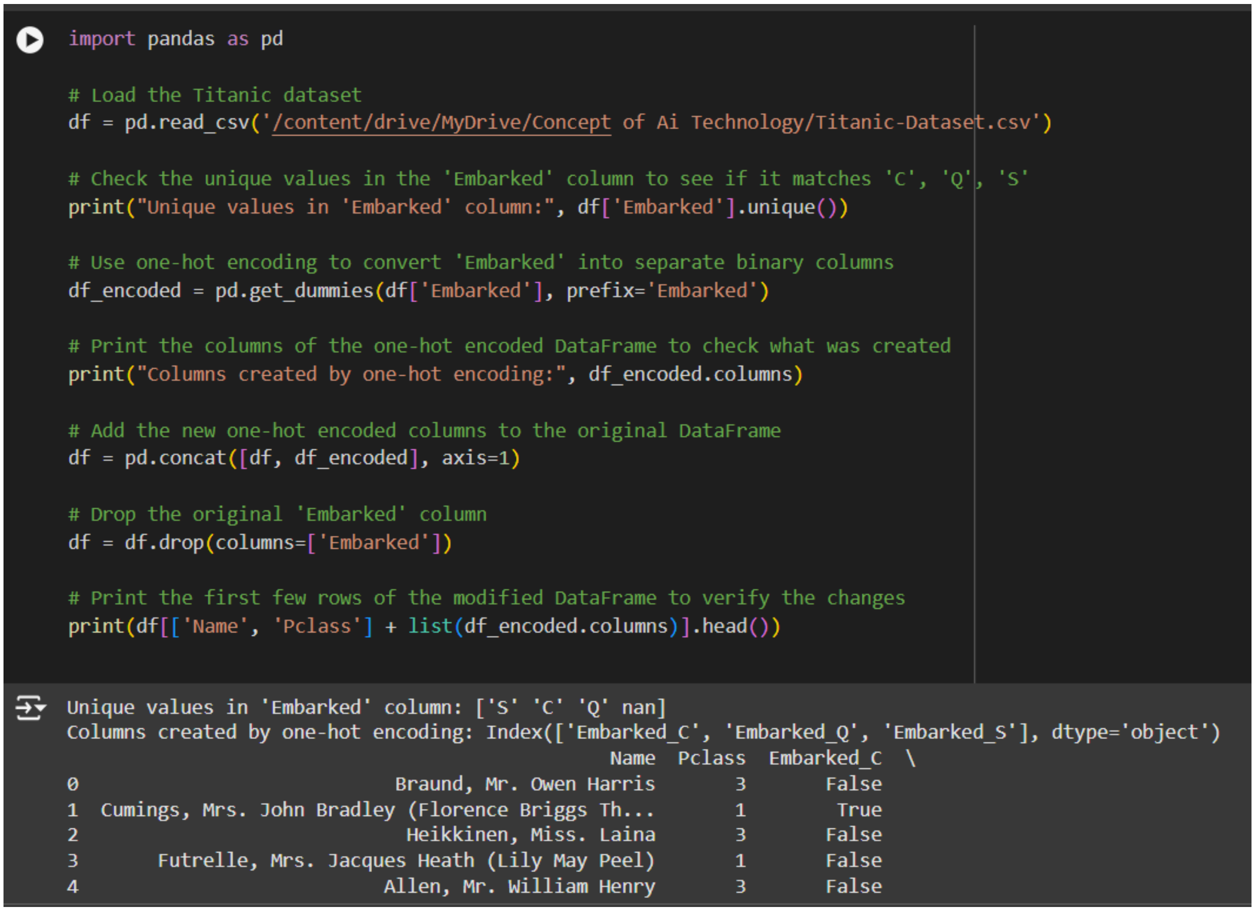


```
#3.
import pandas as pd

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Check the unique values in the 'Embarked' column to see if it matches 'C', 'Q', 'S'
print(df['Embarked'].unique())

# Use one-hot encoding to convert 'Embarked' into separate binary columns
df_encoded = pd.get_dummies(df['Embarked'], prefix='Embarked')

# Add the new one-hot encoded columns to the original DataFrame
df = pd.concat([df, df_encoded], axis=1)

# Drop the original 'Embarked' column
df = df.drop(columns=['Embarked'])

# Print the first few rows of the modified DataFrame to verify the changes
print(df[['Name', 'Pclass'] + list(df_encoded.columns)].head())
```

```
['S' 'C' 'Q' nan]
                                                Name  Pclass  Embarked_C  \
0                            Braund, Mr. Owen Harris       3       False
1  Cumings, Mrs. John Bradley (Florence Briggs Th...       1        True
2                             Heikkinen, Miss. Laina       3       False
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)       1       False
4                           Allen, Mr. William Henry       3       False

   Embarked_Q  Embarked_S
0       False        True
1       False       False
```

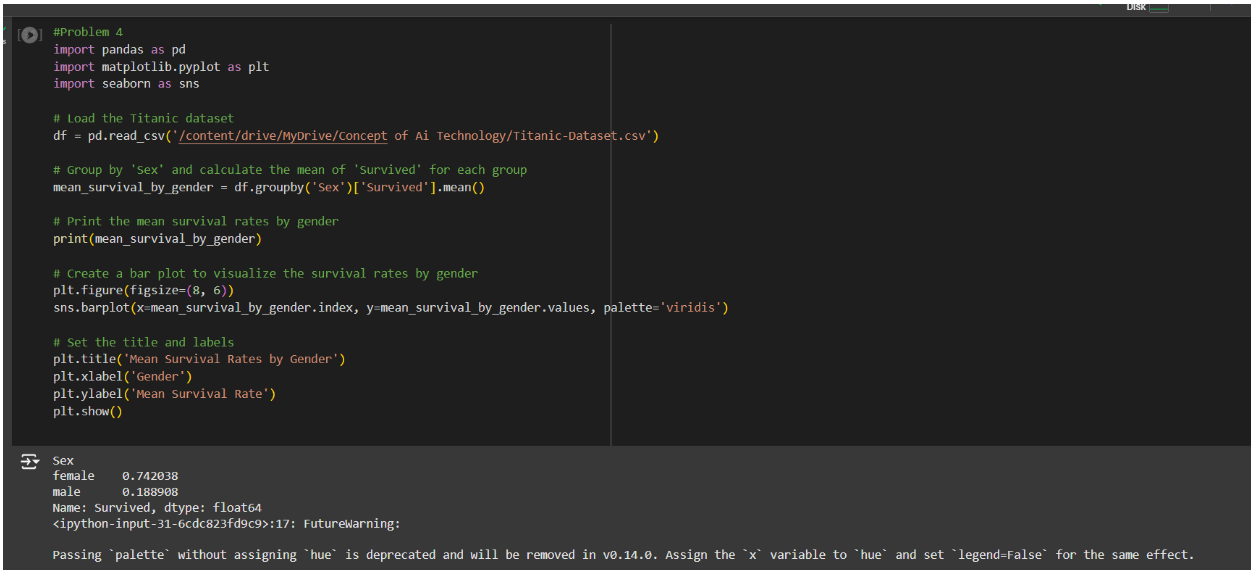


```
['S' 'C' 'Q' nan]
                                                Name  Pclass  Embarked_C  \
0                            Braund, Mr. Owen Harris       3       False
1  Cumings, Mrs. John Bradley (Florence Briggs Th...       1        True
2                             Heikkinen, Miss. Laina       3       False
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)       1       False
4                           Allen, Mr. William Henry       3       False

   Embarked_Q  Embarked_S
0       False        True
1       False       False
2       False        True
3       False        True
4       False        True
```



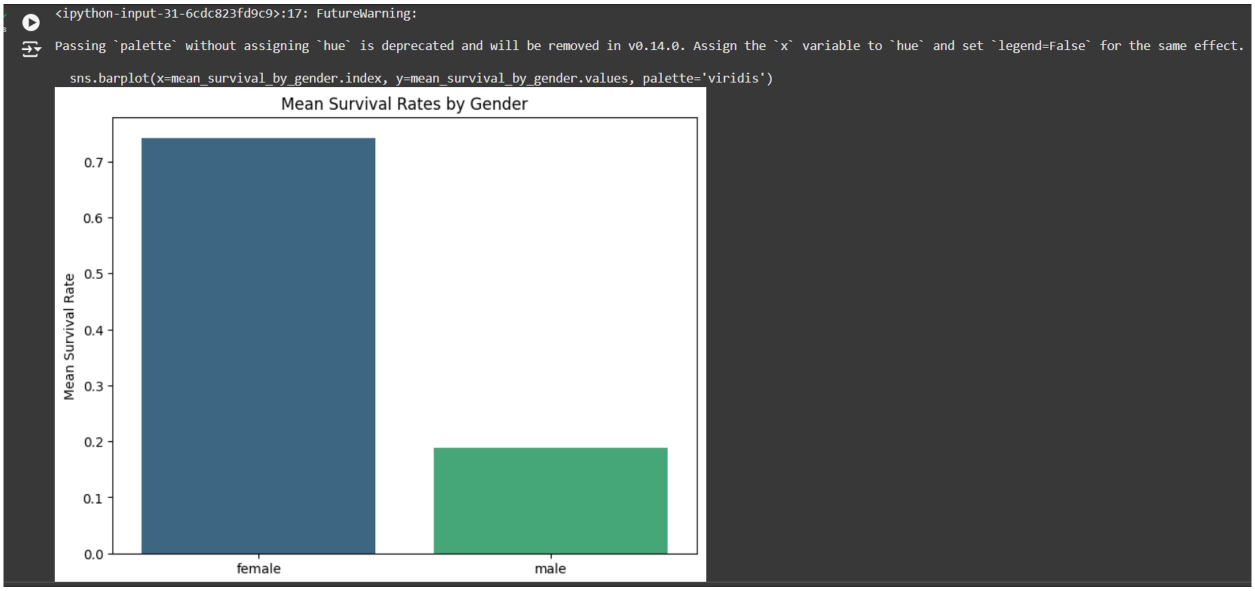
```
import pandas as pd

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Check the unique values in the 'Embarked' column to see if it matches 'C', 'Q', 'S'
print("Unique values in 'Embarked' column:", df['Embarked'].unique())

# Use one-hot encoding to convert 'Embarked' into separate binary columns
df_encoded = pd.get_dummies(df['Embarked'], prefix='Embarked')

# Print the columns of the one-hot encoded DataFrame to check what was created
print("Columns created by one-hot encoding:", df_encoded.columns)

# Add the new one-hot encoded columns to the original DataFrame
df = pd.concat([df, df_encoded], axis=1)

# Drop the original 'Embarked' column
df = df.drop(columns=['Embarked'])

# Print the first few rows of the modified DataFrame to verify the changes
print(df[['Name', 'Pclass'] + list(df_encoded.columns)].head())
```

```
Unique values in 'Embarked' column: ['S' 'C' 'Q' nan]
Columns created by one-hot encoding: Index(['Embarked_C', 'Embarked_Q', 'Embarked_S'], dtype='object')
                                                    Name  Pclass  Embarked_C  \
0                                Braund, Mr. Owen Harris       3       False
1      Cumings, Mrs. John Bradley (Florence Briggs Th...       1        True
2                                 Heikkinen, Miss. Laina       3       False
3           Futrelle, Mrs. Jacques Heath (Lily May Peel)       1       False
4                               Allen, Mr. William Henry       3       False
```

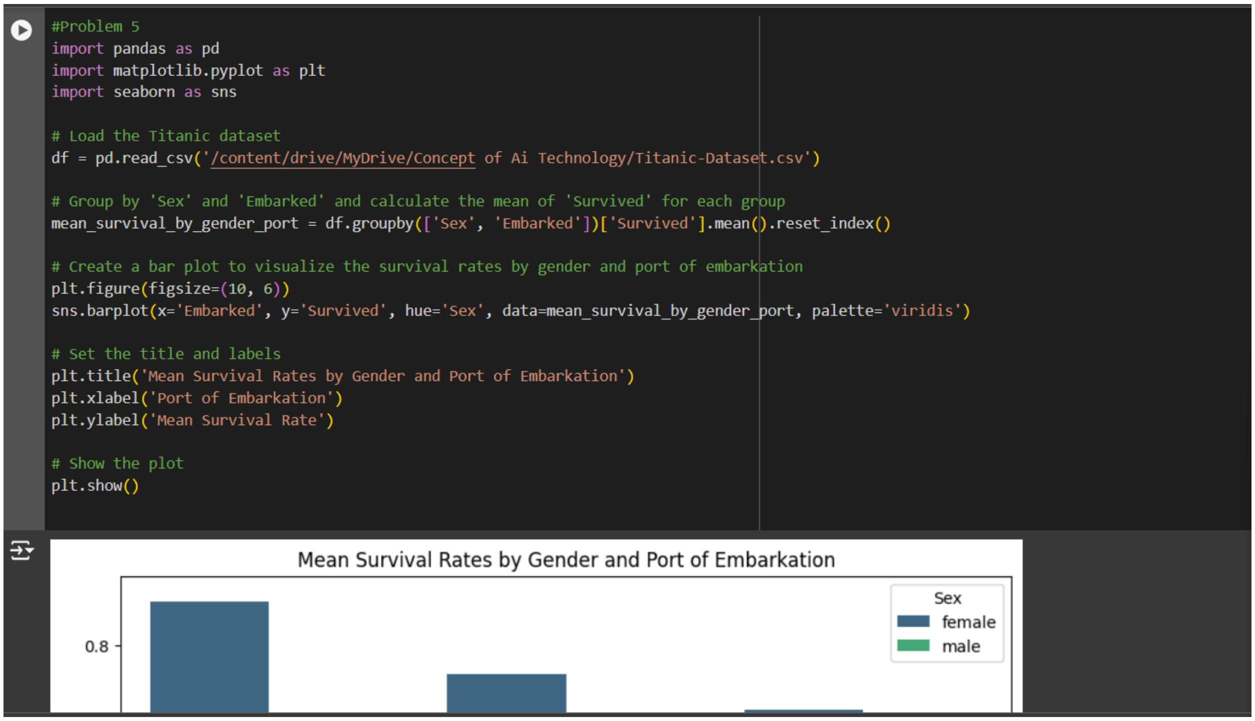


```
#Problem 4
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Group by 'Sex' and calculate the mean of 'Survived' for each group
mean_survival_by_gender = df.groupby('Sex')['Survived'].mean()

# Print the mean survival rates by gender
print(mean_survival_by_gender)

# Create a bar plot to visualize the survival rates by gender
plt.figure(figsize=(8, 6))
sns.barplot(x=mean_survival_by_gender.index, y=mean_survival_by_gender.values, palette='viridis')

# Set the title and labels
plt.title('Mean Survival Rates by Gender')
plt.xlabel('Gender')
plt.ylabel('Mean Survival Rate')
plt.show()
```

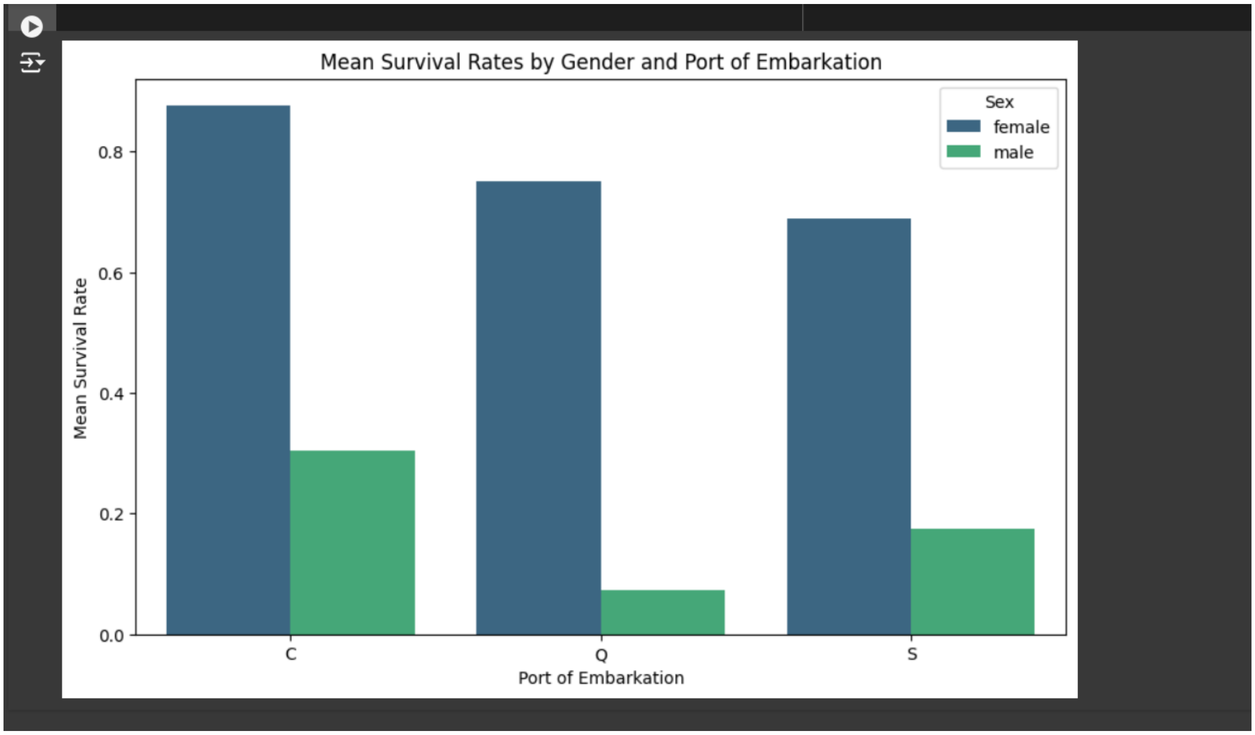
```
Sex
female    0.742038
male      0.188908
Name: Survived, dtype: float64
<ipython-input-31-6cdc823fd9c9>:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.
```



```
<ipython-input-31-6cdc823fd9c9>:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=mean_survival_by_gender.index, y=mean_survival_by_gender.values, palette='viridis')
```



```python
#Problem 5
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the Titanic dataset
df = pd.read_csv('/content/drive/MyDrive/Concept of Ai Technology/Titanic-Dataset.csv')

# Group by 'Sex' and 'Embarked' and calculate the mean of 'Survived' for each group
mean_survival_by_gender_port = df.groupby(['Sex', 'Embarked'])['Survived'].mean().reset_index()

# Create a bar plot to visualize the survival rates by gender and port of embarkation
plt.figure(figsize=(10, 6))
sns.barplot(x='Embarked', y='Survived', hue='Sex', data=mean_survival_by_gender_port, palette='viridis')

# Set the title and labels
plt.title('Mean Survival Rates by Gender and Port of Embarkation')
plt.xlabel('Port of Embarkation')
plt.ylabel('Mean Survival Rate')

# Show the plot
plt.show()
```


Mean Survival Rates by Gender and Port of Embarkation
Sex
female
male
Mean Survival Rate
0.0
0.2
0.4
0.6
0.8
C
Q
S
Port of Embarkation