1. You have a pandas DataFrame df with columns A and B. Column A has some missing values. Write code to replace missing values in column A with the mean value of that column.

In [143…

```python
import pandas as pd

num_df = pd.DataFrame({
    'A': [1, 2, None,  4, 5, None],
     'B': [10, 20, 30, 40, 50, 60]
 })
```

In [145…

```python
# Printing the rows with the head function
num_df.head(6)
```

Out[145]:

	A	B
0	1.0	10
1	2.0	20
2	NaN	30
3	4.0	40
4	5.0	50
5	NaN	60

In [147…

```python
# Calculate the mean of column A, ignoring NaN values
mean_value = num_df['A'].mean()

# Replace missing values in column A with the mean value
num_df['A'].fillna(mean_value, inplace=True)

# Display the DataFrame after replacing NaN values
print(num_df)
```

```
     A   B
0  1.0  10
1  2.0  20
2  3.0  30
3  4.0  40
4  5.0  50
5  3.0  60
```

2. Given the same DataFrame df, write code to remove any rows that contain at least one missing value.

In [150…

```python
import pandas as pd

num_df = pd.DataFrame({
    'A': [1, 2, None,  4, 5, None],
     'B': [10, 20, 30, 40, 50, 60]
 })
```

In [152…

```python
# Printing df
num_df.head(6)
```

Out[152]:

	A	B
0	1.0	10

1	2.0	20
2	NaN	30
3	4.0	40
4	5.0	50
5	NaN	60

In [154…
```
# Remove rows with any missing values
df_cleaned = num_df.dropna()

# Display the cleaned DataFrame
print(df_cleaned)
```

```
     A   B
0  1.0  10
1  2.0  20
3  4.0  40
4  5.0  50
```

3. You have a pandas DataFrame df with columns Product and Sales. Write code to compute the total sales for each unique product.

In [157…
```
import pandas as pd

# Product sales DataFrame
prod_df = pd.DataFrame({
    'Product': ['Heater', 'Bioler', 'Heater', 'Cooler', 'Bioler', 'Heater'],
     'Sales': [100, 150, 200, 300, 250, 50]
 })
```

In [159…
```
# Compute the total sales for each unique product
total_sales_per_product = prod_df.groupby('Product')['Sales'].sum()

# Display the result
print(total_sales_per_product)
```

```
Product
Bioler    400
Cooler    300
Heater    350
Name: Sales, dtype: int64
```

4. Write a code snippet to read data from a CSV file named data.csv into a DataFrame df_csv.

In [162…
```
import pandas as pd

# Read data.csv file with dataframe name df_csv

df_csv = pd.read_csv('data.csv')
```

In [164…
```
# Print dataframe with first 5 rows with head()

df_csv.head()
```

Out[164]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	co
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	

2	84300903	M	19.69	21.25	130.00	1203.0	0.10960
3	84348301	M	11.42	20.38	77.58	386.1	0.14250
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030

5 rows × 33 columns

In [166…

```
# Print dataframe with print function

print(df_csv)
```

```
           id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0      842302         M        17.99         10.38          122.80     1001.0
1      842517         M        20.57         17.77          132.90     1326.0
2    84300903         M        19.69         21.25          130.00     1203.0
3    84348301         M        11.42         20.38           77.58      386.1
4    84358402         M        20.29         14.34          135.10     1297.0
..        ...       ...          ...           ...             ...        ...
564    926424         M        21.56         22.39          142.00     1479.0
565    926682         M        20.13         28.25          131.20     1261.0
566    926954         M        16.60         28.08          108.30      858.1
567    927241         M        20.60         29.33          140.10     1265.0
568     92751         B         7.76         24.54           47.92      181.0

     smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0            0.11840           0.27760         0.30010              0.14710
1            0.08474           0.07864         0.08690              0.07017
2            0.10960           0.15990         0.19740              0.12790
3            0.14250           0.28390         0.24140              0.10520
4            0.10030           0.13280         0.19800              0.10430
..               ...               ...             ...                  ...
564          0.11100           0.11590         0.24390              0.13890
565          0.09780           0.10340         0.14400              0.09791
566          0.08455           0.10230         0.09251              0.05302
567          0.11780           0.27700         0.35140              0.15200
568          0.05263           0.04362         0.00000              0.00000

     ...  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
0    ...          17.33           184.60      2019.0           0.16220
1    ...          23.41           158.80      1956.0           0.12380
2    ...          25.53           152.50      1709.0           0.14440
3    ...          26.50            98.87       567.7           0.20980
4    ...          16.67           152.20      1575.0           0.13740
..   ...            ...              ...         ...               ...
564  ...          26.40           166.10      2027.0           0.14100
565  ...          38.25           155.00      1731.0           0.11660
566  ...          34.12           126.70      1124.0           0.11390
567  ...          39.42           184.60      1821.0           0.16500
568  ...          30.37            59.16       268.6           0.08996

     compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
0              0.66560           0.7119                0.2654          0.4601
1              0.18660           0.2416                0.1860          0.2750
2              0.42450           0.4504                0.2430          0.3613
3              0.86630           0.6869                0.2575          0.6638
4              0.20500           0.4000                0.1625          0.2364
..                 ...              ...                   ...             ...
564            0.21130           0.4107                0.2216          0.2060
565            0.19220           0.3215                0.1628          0.2572
566            0.30940           0.3403                0.1418          0.2218
567            0.86810           0.9387                0.2650          0.4087
568            0.06444           0.0000                0.0000          0.2871
```

```
     fractal_dimension_worst  Unnamed: 32
0                    0.11890          NaN
1                    0.08902          NaN
2                    0.08758          NaN
3                    0.17300          NaN
4                    0.07678          NaN
..                       ...          ...
564                  0.07115          NaN
565                  0.06637          NaN
566                  0.07820          NaN
567                  0.12400          NaN
568                  0.07039          NaN

[569 rows x 33 columns]
```

5. Write code to load the first sheet of an Excel file named data.xlsx into a DataFrame df_excel.

In [169…

```
import pandas as pd

# Read data.xlsx file with dataframe name df_excel

df_excel = pd.read_excel('data.xlsx')
```

In [171…

```
# Print dataframe with first 5 rows with head()

df_excel.head()
```

Out[171]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [173…

```
# print the df_excel dataframe

print(df_excel)
```

```
     age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0     63    1   3       145   233    1        0      150      0      2.3
1     37    1   2       130   250    0        1      187      0      3.5
2     41    0   1       130   204    0        0      172      0      1.4
3     56    1   1       120   236    0        1      178      0      0.8
4     57    0   0       120   354    0        1      163      1      0.6
..   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
298   57    0   0       140   241    0        1      123      1      0.2
299   45    1   3       110   264    0        1      132      0      1.2
300   68    1   0       144   193    1        1      141      0      3.4
301   57    1   0       130   131    0        1      115      1      1.2
302   57    0   1       130   236    0        0      174      0      0.0

     slope  ca  thal  target
0        0   0     1       1
1        0   0     2       1
2        2   0     2       1
3        2   0     2       1
4        2   0     2       1
..     ...  ..   ...     ...
```

```
298       1    0      3        0
299       1    0      3        0
300       1    2      3        0
301       1    1      3        0
302       1    1      2        0

[303 rows x 14 columns]
```

6. You have a 2D array of data containing numerical features. Write code to standardize these features using scikit-learn's StandardScaler.

In [176…

```
# import the required modules
import numpy as np
from sklearn.preprocessing import StandardScaler

# Example 2D array of data (rows are samples, columns are features)
num_data = np.array([[9, 8, 7],
                 [6, 5, 4],
                 [3, 2, 1]])
```

In [178…

```
num_data
```

Out[178]:

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

In [180…

```
# Initialize the StandardScaler
scaler = StandardScaler()

# Fit the scaler to the data and transform it
standardized_data = scaler.fit_transform(num_data)

# Display the standardized data
# Each column (feature) now has a mean of 0 and a standard deviation of 1.
print(standardized_data)
```

```
[[ 1.22474487  1.22474487  1.22474487]
 [ 0.          0.          0.        ]
 [-1.22474487 -1.22474487 -1.22474487]]
```

In [ ]: