EDS ACTIVITY NO.1

NAME: SAKSHI BARAHTE
DIV: CS5
ROLL NO.: CS5-61

PRN: 202401100110
SUBJECT: EDS

test.py

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```python
import kagglehub

# Download latest version
path = kagglehub.dataset_download("fnbalves/paper-reviews-data-set")

print("Path to dataset files:", path)

import pandas as pd
import numpy as np

# --- Create a Sample Dataset ---
# This is a dummy dataset to demonstrate the solutions.
# In a real scenario, you would load your data from a file (e.g., CSV).
# Make sure the column names in the code match your actual dataset columns.

data = {
    'review_id': range(1, 21),
    'paper_id': [101, 102, 101, 103, 102, 101, 103, 104, 102, 104, 105, 101, 103, 105, 102, 104, 101, 105, 103, 104],
    'reviewer_id': [201, 202, 203, 201, 203, 202, 201, 204, 204, 203, 205, 201, 202, 205, 203, 204, 202, 205, 201, 203],
    'review_text': [
        'Good paper, clear methodology.',
        'Needs significant improvements in writing.',
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Sample Data Head:
   review_id  paper_id  reviewer_id  ...  recommendation  reviewer_expertise review_length
0          1       101          201  ...          Accept                High            30
1          2       102          202  ...  Major Revision              Medium            42
2          3       101          203  ...          Accept                High            36
3          4       103          201  ...  Minor Revision              Medium            42
4          5       102          203  ...          Accept                 Low            35

[5 rows x 9 columns]

Sample Data Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
```

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
            'High', 'High', 'Medium', 'Low', 'Medium',
            'High', 'Medium', 'High', 'Medium', 'Low'
        ]
    }

    df = pd.DataFrame(data)

    # Add a column for review length (number of characters)
    df['review_length'] = df['review_text'].str.len()

    # Introduce some missing values for demonstration
    df.loc[[2, 5, 11], 'rating'] = np.nan
    df.loc[[7, 14], 'reviewer_expertise'] = np.nan


    print("--- Paper Review Data Analysis ---")
    print("\nSample Data Head:")
    print(df.head())
    print("\nSample Data Info:")
    df.info()
    print("-" * 40)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   review_id          20 non-null     int64
 1   paper_id           20 non-null     int64
 2   reviewer_id        20 non-null     int64
 3   review_text        20 non-null     object
 4   rating             17 non-null     float64
 5   submission_date    20 non-null     datetime64[ns]
 6   recommendation     20 non-null     object
 7   reviewer_expertise  18 non-null     object
 8   review_length      20 non-null     int64
dtypes: datetime64[ns](1), float64(1), int64(4), object(3)
memory usage: 1.5+ KB
```

PROBLEM STATEMENTS AND ANSWERS 1 AND 2:

```
81
82  # Problem 1: Display the first 5 rows of the dataset.
83  print("\nProblem 1: Display the first 5 rows of the dataset.")
84  print("Solution:")
85  print(df.head())
86  print("-" * 40)
87
88  # Problem 2: Get a summary of the numerical columns in the dataset.
89  print("\nProblem 2: Get a summary of the numerical columns in the dataset.")
90  print("Solution:")
91  print(df.describe())
92  print("-" * 40)
93
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 1: Display the first 5 rows of the dataset.
Solution:
   review_id  paper_id  reviewer_id  ...  recommendation  reviewer_expertise review_length
0          1       101          201  ...          Accept                High            30
1          2       102          202  ...  Major Revision              Medium            42
2          3       101          203  ...          Accept                High            36
3          4       103          201  ...  Minor Revision              Medium            42
4          5       102          203  ...          Accept                 Low            35

[5 rows x 9 columns]
----------------------------------------

Problem 2: Get a summary of the numerical columns in the dataset.
Solution:
       review_id   paper_id  reviewer_id     rating      submission_date  review_length
count   20.00000   20.00000    20.000000  17.000000                   20      20.000000
mean    10.50000  102.80000   202.750000   3.470588  2023-03-06 03:36:00      34.950000
min      1.00000  101.00000   201.000000   2.000000  2023-01-15 00:00:00      21.000000
25%      5.75000  101.75000   201.750000   3.000000  2023-02-08 18:00:00      30.000000
50%     10.50000  103.00000   203.000000   4.000000  2023-03-12 12:00:00      35.500000
75%     15.25000  104.00000   204.000000   4.000000  2023-04-02 00:00:00      42.000000
max     20.00000  105.00000   205.000000   5.000000  2023-04-15 00:00:00      45.000000
std      5.91608    1.43637     1.409554   1.067570                  NaN       7.570545
```

PROBLEM STATEMENTS AND ANSWERS 3, 4 AND 5:

```python
# Problem 3: Find the total number of reviews in the dataset.
print("\nProblem 3: Find the total number of reviews in the dataset.")
print("Solution:")
total_reviews = len(df)
print(f"Total number of reviews: {total_reviews}")
print("-" * 40)

# Problem 4: Count the number of unique papers reviewed.
print("\nProblem 4: Count the number of unique papers reviewed.")
print("Solution:")
unique_papers = df['paper_id'].nunique()
print(f"Number of unique papers reviewed: {unique_papers}")
print("-" * 40)

# Problem 5: Count the number of unique reviewers.
print("\nProblem 5: Count the number of unique reviewers.")
print("Solution:")
unique_reviewers = df['reviewer_id'].nunique()
print(f"Number of unique reviewers: {unique_reviewers}")
print("-" * 40)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 3: Find the total number of reviews in the dataset.
Solution:
Total number of reviews: 20
----------------------------------------

Problem 4: Count the number of unique papers reviewed.
Solution:
Number of unique papers reviewed: 5
----------------------------------------

Problem 5: Count the number of unique reviewers.
Solution:
Number of unique reviewers: 5
```

PROBLEM STATEMENTS AND ANSWERS 6 AND 7:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
115  # Problem 6: Calculate the average rating given across all reviews (excluding NaNs).
116  print("\nProblem 6: Calculate the average rating given across all reviews (excluding NaNs).")
117  print("Solution:")
118  average_rating = df['rating'].mean()
119  print(f"Average rating: {average_rating:.2f}")
120  print("-" * 40)
121
122  # Problem 7: Find the paper with the highest average rating.
123  print("\nProblem 7: Find the paper with the highest average rating.")
124  print("Solution:")
125  # Calculate average rating per paper, drop papers with no valid ratings
126  avg_rating_per_paper = df.groupby('paper_id')['rating'].mean().dropna()
127  # Find the paper(s) with the maximum average rating
128  if not avg_rating_per_paper.empty:
129      highest_rated_paper_id = avg_rating_per_paper.idxmax()
130      highest_avg_rating = avg_rating_per_paper.max()
131      print(f"Paper(s) with the highest average rating (ID: Average Rating):")
132      # Handle cases where multiple papers have the same highest rating
133      highest_rated_papers = avg_rating_per_paper[avg_rating_per_paper == highest_avg_rating]
134      print(highest_rated_papers)
135  else:
136      print("No papers with valid ratings found.")
137  print("-" * 40)
138
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 6: Calculate the average rating given across all reviews (excluding NaNs).
Solution:
Average rating: 3.47
----------------------------------------

Problem 7: Find the paper with the highest average rating.
Solution:
Paper(s) with the highest average rating (ID: Average Rating):
paper_id
103    3.75
Name: rating, dtype: float64
```

PROBLEM STATEMENTS AND ANSWERS 8 AND 9:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```python
# Problem 8: Determine the distribution of recommendations ('Accept', 'Reject', etc.).
print("\nProblem 8: Determine the distribution of recommendations.")
print("Solution:")
recommendation_distribution = df['recommendation'].value_counts()
print("Distribution of recommendations:")
print(recommendation_distribution)
print("-" * 40)

# Problem 9: Filter the dataset to show only reviews with a rating of 5.
print("\nProblem 9: Filter the dataset to show only reviews with a rating of 5.")
print("Solution:")
rating_5_reviews = df[df['rating'] == 5]
print("Reviews with a rating of 5:")
print(rating_5_reviews)
print("-" * 40)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 8: Determine the distribution of recommendations.
Solution:
Distribution of recommendations:
recommendation
Accept           9
Minor Revision   5
Major Revision   3
Reject           3
Name: count, dtype: int64
----------------------------------------

Problem 9: Filter the dataset to show only reviews with a rating of 5.
Solution:
Reviews with a rating of 5:
    review_id  paper_id  reviewer_id  ... recommendation  reviewer_expertise review_length
7           8       104          204  ...         Accept                 NaN            44
15         16       104          204  ...         Accept                High            30
18         19       103          201  ...         Accept              Medium            21

[3 rows x 9 columns]
----------------------------------------
```

PROBLEM STATEMENTS AND ANSWERS 10 AND 11:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
155  # Problem 10: Find the average rating for each reviewer expertise level.
156  print("\nProblem 10: Find the average rating for each reviewer expertise level.")
157  print("Solution:")
158  avg_rating_by_expertise = df.groupby('reviewer_expertise')['rating'].mean()
159  print("Average rating by reviewer expertise level:")
160  print(avg_rating_by_expertise)
161  print("-" * 40)
162
163  # Problem 11: Identify the reviewer who has submitted the most reviews.
164  print("\nProblem 11: Identify the reviewer who has submitted the most reviews.")
165  print("Solution:")
166  # Count reviews per reviewer and find the reviewer with the max count
167  reviews_per_reviewer = df['reviewer_id'].value_counts()
168  if not reviews_per_reviewer.empty:
169      most_active_reviewer_id = reviews_per_reviewer.idxmax()
170      most_reviews_count = reviews_per_reviewer.max()
171      print(f"Reviewer who submitted the most reviews: Reviewer ID {most_active_reviewer_id} ({most_reviews_count} reviews)")
172  else:
173      print("No reviews found.")
174  print("-" * 40)
175
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 10: Find the average rating for each reviewer expertise level.
Solution:
Average rating by reviewer expertise level:
reviewer_expertise
High      4.000000
Low       3.000000
Medium    3.142857
Name: rating, dtype: float64
----------------------------------------

Problem 11: Identify the reviewer who has submitted the most reviews.
Solution:
Reviewer who submitted the most reviews: Reviewer ID 201 (5 reviews)
----------------------------------------
```

PROBLEM STATEMENTS AND ANSWERS 12 AND 13:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
175
176  # Problem 12: Calculate the length of each review text (number of characters) and add it as a new column 'review_length'.
177  # Note: This was already done during dataset creation for convenience, but the code is shown here.
178  print("\nProblem 12: Calculate the length of each review text and add it as a new column 'review_length'.")
179  print("Solution: (Column 'review_length' already added during setup)")
180  # df['review_length'] = df['review_text'].str.len() # This line would add the column if it didn't exist
181  print(df[['review_text', 'review_length']].head())
182  print("-" * 40)
183
184  # Problem 13: Find the average review length.
185  print("\nProblem 13: Find the average review length.")
186  print("Solution:")
187  average_review_length = df['review_length'].mean()
188  print(f"Average review length (character count): {average_review_length:.2f}")
189  print("-" * 40)
190
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
----------------------------------------

Problem 12: Calculate the length of each review text and add it as a new column 'review_length'.
Solution: (Column 'review_length' already added during setup)
                              review_text  review_length
0             Good paper, clear methodology.            30
1  Needs significant improvements in writing.            42
2        Excellent contribution to the field.            36
3  Minor issues with figures, otherwise good.            42
4         Acceptable work, but lacks novelty.            35
----------------------------------------

Problem 13: Find the average review length.
Solution:
Average review length (character count): 34.95
----------------------------------------
```

PROBLEM STATEMENT AND ANSWER 14:

test.py

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```python

# Problem 14: Determine the paper(s) that received the most reviews.
print("\nProblem 14: Determine the paper(s) that received the most reviews.")
print("Solution:")
reviews_count_per_paper = df['paper_id'].value_counts()
if not reviews_count_per_paper.empty:
    most_reviewed_paper_id = reviews_count_per_paper.idxmax()
    most_reviews_count = reviews_count_per_paper.max()
    print(f"Paper(s) that received the most reviews (ID: Number of Reviews):")
    # Handle cases where multiple papers received the same highest number of reviews
    most_reviewed_papers = reviews_count_per_paper[reviews_count_per_paper == most_reviews_count]
    print(most_reviewed_papers)
else:
    print("No papers found.")
print("-" * 40)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
----------------------------------------

Problem 14: Determine the paper(s) that received the most reviews.
Solution:
Paper(s) that received the most reviews (ID: Number of Reviews):
paper_id
101    5
Name: count, dtype: int64
----------------------------------------
```

PROBLEM STATEMENT AND ANSWER 15:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
206   # Problem 15: Filter reviews submitted after a specific date (e.g., '2023-03-01').
207   print("\nProblem 15: Filter reviews submitted after a specific date (e.g., '2023-03-01').")
208   print("Solution:")
209   # Ensure 'submission_date' is in datetime format (already done during setup)
210   # df['submission_date'] = pd.to_datetime(df['submission_date'])
211   reviews_after_date = df[df['submission_date'] > '2023-03-01']
212   print("Reviews submitted after 2023-03-01:")
213   print(reviews_after_date)
214   print("-" * 40)
215
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 15: Filter reviews submitted after a specific date (e.g., '2023-03-01').
Solution:
Reviews submitted after 2023-03-01:
    review_id  paper_id  reviewer_id  ...  recommendation  reviewer_expertise review_length
8           9       102          204  ...          Accept                 Low            40
9          10       104          203  ...          Reject              Medium            22
10         11       105          205  ...  Major Revision                High            43
11         12       101          201  ...          Accept                High            45
12         13       103          202  ...  Minor Revision              Medium            40
13         14       105          205  ...          Reject                 Low            27
14         15       102          203  ...  Minor Revision                 NaN            33
15         16       104          204  ...          Accept                High            30
16         17       101          202  ...  Minor Revision              Medium            24
17         18       105          205  ...  Major Revision                High            39
18         19       103          201  ...          Accept              Medium            21
19         20       104          203  ...          Reject                 Low            34

[12 rows x 9 columns]
----------------------------------------
```

PROBLEM STATEMENTS AND ANSWERS 16 AND 17:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
216  # Problem 16: Count the number of reviews for each paper.
217  print("\nProblem 16: Count the number of reviews for each paper.")
218  print("Solution:")
219  reviews_count_per_paper_sorted = df['paper_id'].value_counts().sort_index()
220  print("Number of reviews for each paper:")
221  print(reviews_count_per_paper_sorted)
222  print("-" * 40)
223
224  # Problem 17: Find the minimum and maximum rating given (excluding NaNs).
225  print("\nProblem 17: Find the minimum and maximum rating given (excluding NaNs).")
226  print("Solution:")
227  min_rating = df['rating'].min()
228  max_rating = df['rating'].max()
229  print(f"Minimum rating: {min_rating}, Maximum rating: {max_rating}")
230  print("-" * 40)
231
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
----------------------------------------

Problem 16: Count the number of reviews for each paper.
Solution:
Number of reviews for each paper:
paper_id
101    5
102    4
103    4
104    4
105    3
Name: count, dtype: int64
----------------------------------------

Problem 17: Find the minimum and maximum rating given (excluding NaNs).
Solution:
Minimum rating: 2.0, Maximum rating: 5.0
----------------------------------------
```

PROBLEM STATEMENT AND ANSWER 18:

C: > Users > Atulya > OneDrive > Desktop > test.py > ...

```
232  # Problem 18: Group reviews by paper and find the average rating and the number of reviews for each paper.
233  print("\nProblem 18: Group reviews by paper and find the average rating and the number of reviews for each paper.")
234  print("Solution:")
235  paper_summary = df.groupby('paper_id').agg(
236      average_rating=('rating', 'mean'),
237      number_of_reviews=('review_id', 'count')
238  )
239  print("Summary (average rating and review count) for each paper:")
240  print(paper_summary)
241  print("-" * 40)
242
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 18: Group reviews by paper and find the average rating and the number of reviews for each paper.
Solution:
Summary (average rating and review count) for each paper:
          average_rating  number_of_reviews
paper_id
101                 3.50                  5
102                 3.50                  4
103                 3.75                  4
104                 3.50                  4
105                 3.00                  3
----------------------------------------
```

PROBLEM STATEMENT AND ANSWER 19:

```
C: > Users > Atulya > OneDrive > Desktop > test.py > ...
243   # Problem 19: Check for missing values in each column.
244   print("\nProblem 19: Check for missing values in each column.")
245   print("Solution:")
246   missing_values = df.isnull().sum()
247   print("Missing values per column:")
248   print(missing_values)
249   print("-" * 40)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

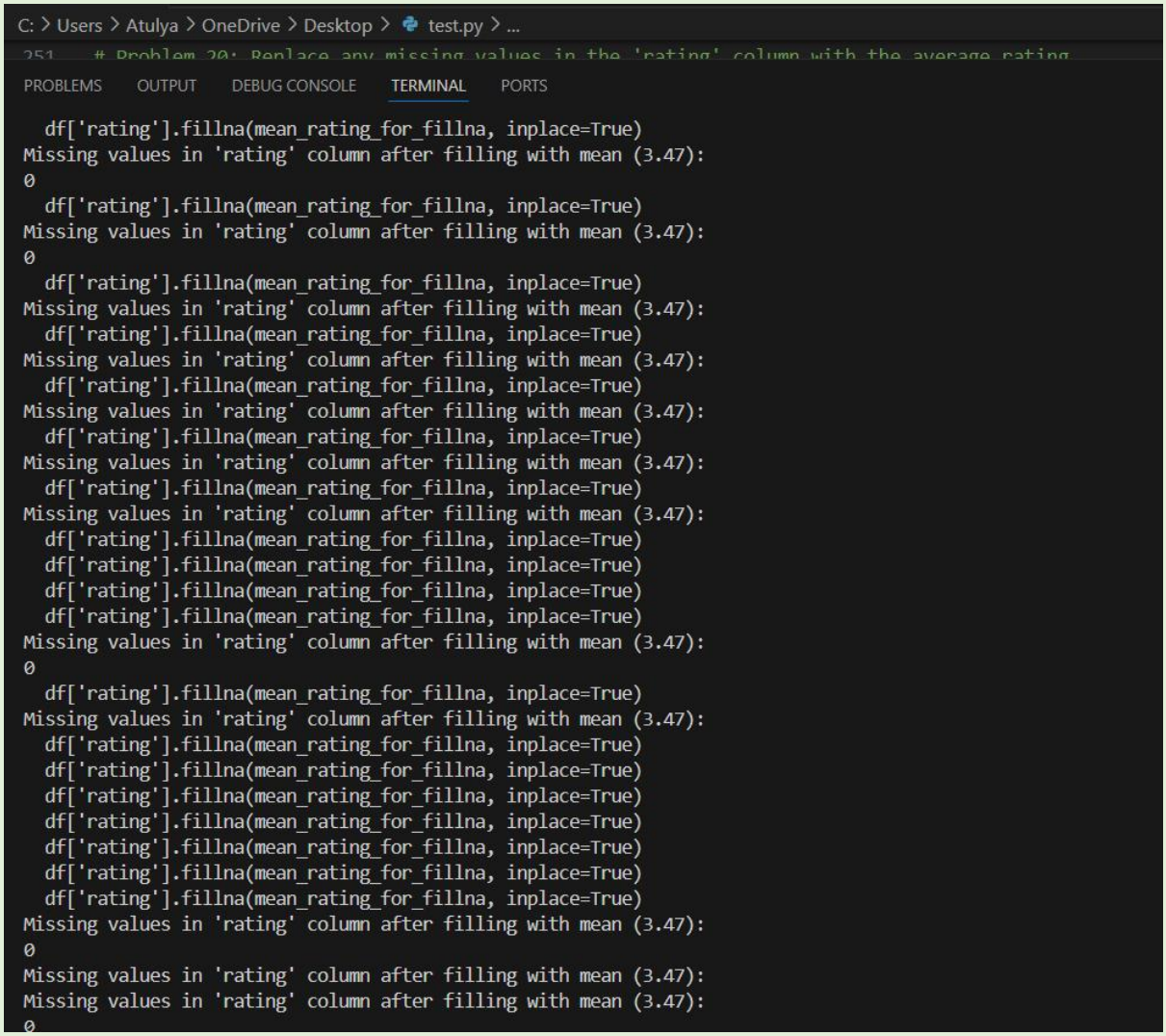
```
Problem 19: Check for missing values in each column.
Solution:
Missing values per column:
review_id             0
paper_id              0
reviewer_id           0
review_text           0
rating                3
submission_date       0
recommendation        0
reviewer_expertise    2
review_length         0
dtype: int64
```

PROBLEM STATEMENT AND ANSWER 20:

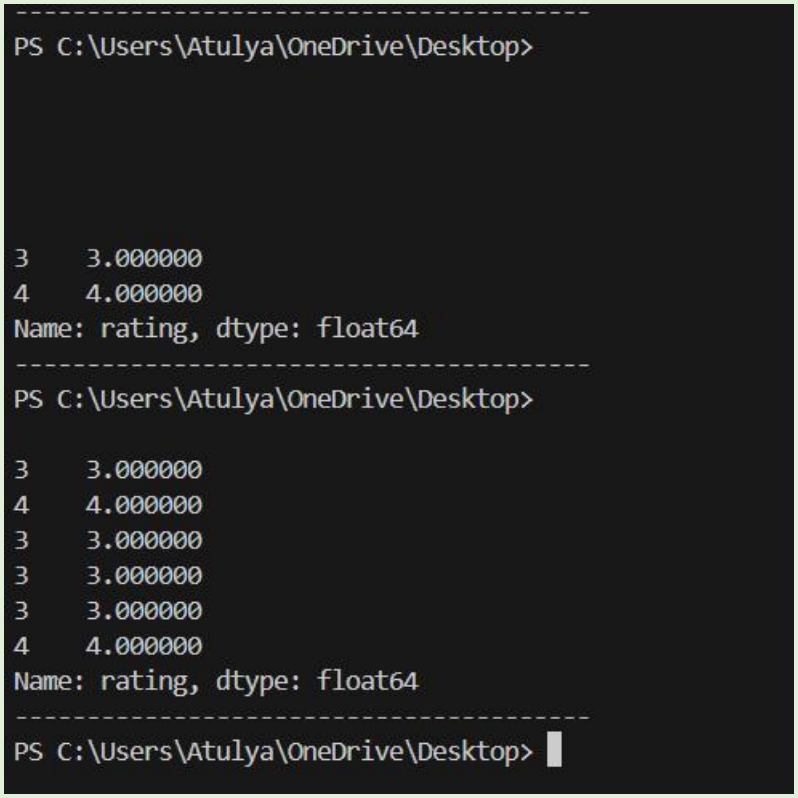
```
C: > Users > Atulya > OneDrive > Desktop > test.py > ...
251   # Problem 20: Replace any missing values in the 'rating' column with the average rating.
252   print("\nProblem 20: Replace any missing values in the 'rating' column with the average rating.")
253   print("Solution:")
254   # Calculate the mean rating excluding NaNs for filling
255   mean_rating_for_fillna = df['rating'].mean()
256   df['rating'].fillna(mean_rating_for_fillna, inplace=True)
257   print(f"Missing values in 'rating' column after filling with mean ({mean_rating_for_fillna:.2f}):")
258   print(df['rating'].isnull().sum())
259   print("\nFirst 5 rows of 'rating' column after filling NaNs:")
260   print(df['rating'].head())
261   print("-" * 40)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Problem 20: Replace any missing values in the 'rating' column with the average rating.
Solution:
C:\Users\Atulya\OneDrive\Desktop\test.py:258: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
ignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values alwa
ys behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(va
lue) instead, to perform the operation inplace on the original object.


  df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0

First 5 rows of 'rating' column after filling NaNs:
  df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0

  df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0
```


C: > Users > Atulya > OneDrive > Desktop > test.py > ...
Problem 20: Replace any missing values in the 'rating' column with the average rating
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
df['rating'].fillna(mean_rating_for_fillna, inplace=True)
Missing values in 'rating' column after filling with mean (3.47):
0
Missing values in 'rating' column after filling with mean (3.47):
Missing values in 'rating' column after filling with mean (3.47):
0

```
First 5 rows of 'rating' column after filling NaNs:
0    4.000000
First 5 rows of 'rating' column after filling NaNs:
0    4.000000
1    2.000000
2    3.470588
0    4.000000
1    2.000000
2    3.470588
1    2.000000
2    3.470588
2    3.470588
3    3.000000
4    4.000000
Name: rating, dtype: float64
3    3.000000
4    4.000000
3    3.000000
3    3.000000
3    3.000000
4    4.000000
Name: rating, dtype: float64
----------------------------------------
3    3.000000
4    4.000000
Name: rating, dtype: float64
3    3.000000
4    4.000000
3    3.000000
3    3.000000
3    3.000000
3    3.000000
3    3.000000
3    3.000000
3    3.000000
4    4.000000
Name: rating, dtype: float64
```


PS C:\Users\Atulya\OneDrive\Desktop>
3 3.000000
4 4.000000
Name: rating, dtype: float64
PS C:\Users\Atulya\OneDrive\Desktop>
3 3.000000
4 4.000000
3 3.000000
3 3.000000
3 3.000000
4 4.000000
Name: rating, dtype: float64
PS C:\Users\Atulya\OneDrive\Desktop>