ACITVITY NO 1

Name: MAYANK DANTRE

PRN: 202401080047

Roll Number: CS6-57

Class: CS6

📚 About the Dataset

The dataset used in this assignment is the **OpinRank Hotel Review Dataset**.
It contains hotel reviews collected from various cities and countries.

Each review entry includes:

- The **Date** when the review was posted
- A **Title** summarizing the review
- The full **Review Text** providing detailed comments

This dataset can be used for various Natural Language Processing (NLP) tasks like:

- Summarization
- Customer feedback mining
- Reviews

✨ Problem Statements

Problem 1: Find the total number of reviews.

```python
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

Generate | create a dataframe with 2 columns and 10 rows | Close

```python
# Problem 1
total_reviews = len(df)
print("Total number of reviews:", total_reviews)
```

OUTPUT

```
Total number of reviews: 4783
```

Problem 2: Find the earliest and latest review dates.

```python
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

Generate | a slider using jupyter widgets | Close

```python
# Problem 2
earliest_date = df['Date'].min()
latest_date = df['Date'].max()
print("Earliest review date:", earliest_date)
print("Latest review date:", latest_date)
```

OUTPUT

```
Earliest review date: NaT
Latest review date: NaT
```

Problem 3: Count the number of reviews posted each year.

```python
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

Generate | randomly select 5 items from a list | Close

```python
# Problem 3
reviews_per_year = df['Year'].value_counts().sort_index()
print("Number of reviews per year:")
print(reviews_per_year)
```

OUTPUT

```
Number of reviews per year:
Series([], Name: count, dtype: int64)
```

Problem 4: Find the average length of review texts (in characters).

- Video Interpolation: Predict what happened in a video between the first and the last frame.

+ Code + Text

```
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

```
# Problem 4
average_length = df['Review'].str.len().mean()
print("Average length of review texts (in characters):", average_length)
```

OUTPUT

```
Average length of review texts (in characters): 48.93581434246289
```

Problem 5: Find the review title that has the maximum number of words.

- Video Interpolation: Predict what happened in a video between the first and the last frame.

```
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

```
# Problem 5
title_word_counts = df['Title'].str.split().map(len)
index_max_title = title_word_counts.idxmax()
print("Title with maximum number of words:", df.loc[index_max_title, 'Title'])
```

OUTPUT

```
Title with maximum number of words: n Cd□h}7□7□.ILv4-s□□□|/□>V1W=.
```

Problem 6: Calculate the number of reviews that mention "taxi" in the review text.

```
[ ]  import pandas as pd
     import numpy as np

     # Load data
     data = []
     with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
         for line in file:
             parts = line.strip().split('\t')
             if len(parts) == 3:
                 data.append(parts)

     # Create DataFrame
     df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

     # Parse 'Date' to datetime
     df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

     # Extract Year and Month
     df['Year'] = df['Date'].dt.year
     df['Month'] = df['Date'].dt.month
```

```
# Problem 6
taxi_mention_count = df['Review'].str.contains('taxi', case=False, na=False).sum()
print("Number of reviews mentioning 'taxi':", taxi_mention_count)
```

OUTPUT

```
Number of reviews mentioning 'taxi': 0
```

Problem 7: Find how many unique review titles there are.

```
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

```
# Problem 7
unique_titles_count = df['Title'].nunique()
print("Number of unique review titles:", unique_titles_count)
```

OUTPUT

Problem 8: Calculate the median number of words per review.

```
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

```
# Problem 8
median_word_count = df['Review'].str.split().map(len).median()
print("Median number of words per review:", median_word_count)
```

OUTPUT

Problem 9: Create a list of all unique words used in the titles.

```
[ ]  import pandas as pd
     import numpy as np

     # Load data
     data = []
     with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
         for line in file:
             parts = line.strip().split('\t')
             if len(parts) == 3:
                 data.append(parts)

     # Create DataFrame
     df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

     # Parse 'Date' to datetime
     df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

     # Extract Year and Month
     df['Year'] = df['Date'].dt.year
     df['Month'] = df['Date'].dt.month
```

```
# Problem 9
unique_title_words = np.unique(np.concatenate(df['Title'].str.split()))
print("List of unique words in all titles:")
print(unique_title_words)
```

OUTPUT

```
List of unique words in all titles:
['' 
 '\x00\x00\x13\x00\x00A\x00\x00\x00hotels/beijing/china_beijing_ho
 '\x00\x00(\x00\x00\x00hotels/beijing/china_beijing_xinyuan_inn]Vn
 ...
 '\U000d8da1B\x03*[a\x06PC>@\x00%8\x15o\u;\\*UJ+Ub\x15s\x07PIn@c¿\\
 '\U000e3e04~U\x01IzIx\' '\U0010f99aDÈ^]']
```

Problem 10: Identify how many reviews are shorter than 100 words.

```
[ ] import pandas as pd
    import numpy as np

    # Load data
    data = []
    with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            parts = line.strip().split('\t')
            if len(parts) == 3:
                data.append(parts)

    # Create DataFrame
    df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

    # Parse 'Date' to datetime
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

    # Extract Year and Month
    df['Year'] = df['Date'].dt.year
    df['Month'] = df['Date'].dt.month
```

```
# Problem 10
short_reviews_count = (df['Review'].str.split().map(len) < 100).sum()
print("Number of reviews with less than 100 words:", short_reviews_count)
```

OUTPUT

```
Number of reviews with less than 100 words: 4783
```

Problem 11: Find the proportion of reviews posted before 2010.

```
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

```
# Problem 11
proportion_before_2010 = (df['Year'] < 2010).mean()
print("Proportion of reviews before 2010:", proportion_before_2010)
```

OUTPUT

```
Proportion of reviews before 2010: 0.0
```

Problem 12: Check the percentage of titles containing the word "excellent".

```python
import pandas as pd
import numpy as np

# Load data
data = []
with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
    for line in file:
        parts = line.strip().split('\t')
        if len(parts) == 3:
            data.append(parts)

# Create DataFrame
df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

# Parse 'Date' to datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract Year and Month
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
```

```python
# Problem 12
excellent_titles_percentage = df['Title'].str.contains('excellent', case=False, na=False).mean() * 100
print("Percentage of titles containing 'excellent':", excellent_titles_percentage)
```

OUTPUT

```
Percentage of titles containing 'excellent': 0.0
```

Problem 13: Determine the month which had the highest number of reviews.

```
[ ] import pandas as pd
    import numpy as np

    # Load data
    data = []
    with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            parts = line.strip().split('\t')
            if len(parts) == 3:
                data.append(parts)

    # Create DataFrame
    df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

    # Parse 'Date' to datetime
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

    # Extract Year and Month
    df['Year'] = df['Date'].dt.year
    df['Month'] = df['Date'].dt.month
```

```
# Problem 13
most_active_month = df['Month'].value_counts().idxmax()
print("Month with the highest number of reviews:", most_active_month)
```

OUTPUT

```
No valid months available in the dataset.
```

Problem 14: Find the review that talks about "internet" the most times.

```
[ ] import pandas as pd
    import numpy as np

    # Load data
    data = []
    with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            parts = line.strip().split('\t')
            if len(parts) == 3:
                data.append(parts)

    # Create DataFrame
    df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

    # Parse 'Date' to datetime
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

    # Extract Year and Month
    df['Year'] = df['Date'].dt.year
    df['Month'] = df['Date'].dt.month
```

```
# Problem 14
internet_counts = df['Review'].str.lower().str.count('internet')
index_most_internet = internet_counts.idxmax()
print("Review mentioning 'internet' the most:")
print(df.loc[index_most_internet, 'Review'])
```

OUTPUT

```
Review mentioning 'internet' the most:
0S7'\7□□sY□□□□□□Z[=!0□S&X)$1a,aw6BNydB(□5□!g□)□□□V(□f=L'□□□_0fGC
```

Problem 15: Create a column with the word count of each review.

```
[ ]  import pandas as pd
     import numpy as np

     # Load data
     data = []
     with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
         for line in file:
             parts = line.strip().split('\t')
             if len(parts) == 3:
                 data.append(parts)

     # Create DataFrame
     df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

     # Parse 'Date' to datetime
     df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

     # Extract Year and Month
     df['Year'] = df['Date'].dt.year
     df['Month'] = df['Date'].dt.month
```

```
# Problem 15
df['Word_Count'] = df['Review'].str.split().map(len)
print("First 5 entries with Word Count column added:")
print(df[['Review', 'Word_Count']].head())
```

OUTPUT

```
First 5 entries with Word Count column added:
                                              Review  Word_Count
0  0S7'\7□□sY□□□□□□Z[=!0□S&X)$1a,aw6BNydB(□5□!g□)...           3
1                          3gh^Ok7(AG□Sæ□w"mk/^□O"□□$n.           1
2                                        □□□,ẌÖ&□□□hZ.-           1
3   ó□h37ж0Eu[SKfcr ~0□,Ⱡ]SiRn̂c*{□□4qn□K&%|□m□□     2
4                                                  □           1
```

Problem 16: Find the mean word count grouped by year.

```
[ ]  import pandas as pd
     import numpy as np

     # Load data
     data = []
     with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
         for line in file:
             parts = line.strip().split('\t')
             if len(parts) == 3:
                 data.append(parts)

     # Create DataFrame
     df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

     # Parse 'Date' to datetime
     df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

     # Extract Year and Month
     df['Year'] = df['Date'].dt.year
     df['Month'] = df['Date'].dt.month
```

```
# Problem 16
mean_words_by_year = df.groupby('Year')['Word_Count'].mean()
print("Mean word count by year:")
print(mean_words_by_year)
```

OUTPUT

```
Mean word count by year:
Series([], Name: Word_Count, dtype: float64)
```

Problem 17: Extract all reviews posted in December.

```
[ ]  import pandas as pd
     import numpy as np

     # Load data
     data = []
     with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
         for line in file:
             parts = line.strip().split('\t')
             if len(parts) == 3:
                 data.append(parts)

     # Create DataFrame
     df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

     # Parse 'Date' to datetime
     df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

     # Extract Year and Month
     df['Year'] = df['Date'].dt.year
     df['Month'] = df['Date'].dt.month
```

```
# Problem 17
december_reviews = df[df['Month'] == 12]
print("All reviews posted in December:")
print(december_reviews)
```

OUTPUT

```
All reviews posted in December:
Empty DataFrame
Columns: [Date, Title, Review, Year, Month, Word_Count]
Index: []
```

Problem 18: Find reviews where the title and body share common words.

```
[ ] import pandas as pd
    import numpy as np

    # Load data
    data = []
    with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            parts = line.strip().split('\t')
            if len(parts) == 3:
                data.append(parts)

    # Create DataFrame
    df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

    # Parse 'Date' to datetime
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

    # Extract Year and Month
    df['Year'] = df['Date'].dt.year
    df['Month'] = df['Date'].dt.month
```

```
# Problem 18
def common_words(title, review):
    title_words = set(str(title).lower().split())
    review_words = set(str(review).lower().split())
    return len(title_words.intersection(review_words)) > 0

df['Common_Words'] = df.apply(lambda x: common_words(x['Title'], x['Review']), axis=1)
common_reviews = df[df['Common_Words']]
print("Reviews where title and review share common words:")
print(common_reviews[['Title', 'Review']])
```

OUTPUT

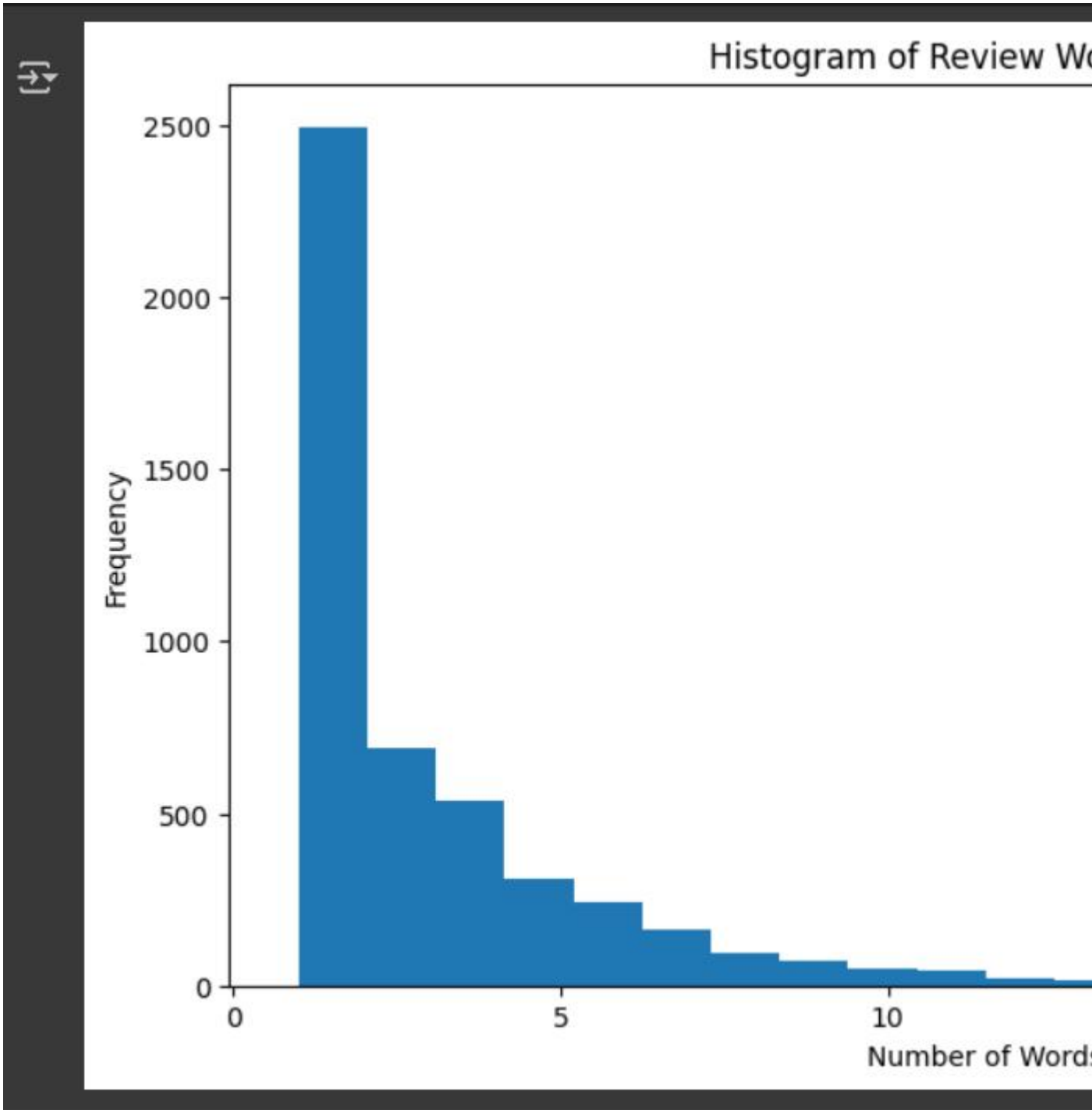
```
Reviews where title and review share common words:
                                                  Title  \
3747  =X□DN□m\kxjr&bN□%.6vG,□□! NS□j□H□□FU}□B[ 0□E

                                                 Review
3747  dlo ’z"□W□G□t□□`□Vi□L‘□Korzx□%eK□UQ□e□-,□
```

Problem 19: Create a histogram of review word counts.

```
[ ] import pandas as pd
    import numpy as np

    # Load data
    data = []
    with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            parts = line.strip().split('\t')
            if len(parts) == 3:
                data.append(parts)

    # Create DataFrame
    df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

    # Parse 'Date' to datetime
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

    # Extract Year and Month
    df['Year'] = df['Date'].dt.year
    df['Month'] = df['Date'].dt.month
```

```
# Problem 19
import matplotlib.pyplot as plt

df['Word_Count'].plot.hist(bins=20, figsize=(10,6))
plt.title('Histogram of Review Word Counts')
plt.xlabel('Number of Words')
plt.ylabel('Frequency')
plt.show()
```

OUTPUT



Problem 20: List the top 5 most common words in all review titles.

```
[ ] import pandas as pd
    import numpy as np

    # Load data
    data = []
    with open('path_to_your_sample_file', 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            parts = line.strip().split('\t')
            if len(parts) == 3:
                data.append(parts)

    # Create DataFrame
    df = pd.DataFrame(data, columns=['Date', 'Title', 'Review'])

    # Parse 'Date' to datetime
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

    # Extract Year and Month
    df['Year'] = df['Date'].dt.year
    df['Month'] = df['Date'].dt.month
```

```
# Problem 20
title_words = ' '.join(df['Title'].dropna()).lower().split()
top_5_words = pd.Series(title_words).value_counts().head(5)
print("Top 5 most common words in titles:")
print(top_5_words)
```

OUTPUT

```
Top 5 most common words in titles:
f    24
b    24
v    23
z    23
m    22
Name: count, dtype: int64
```