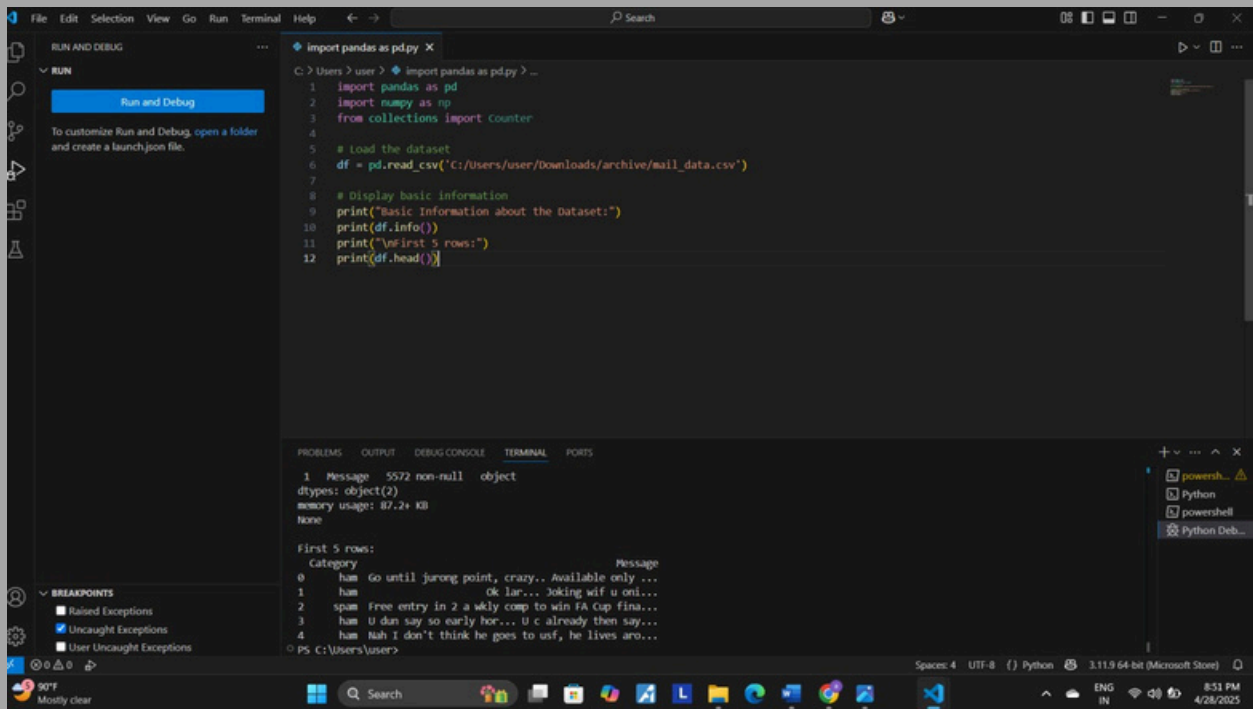
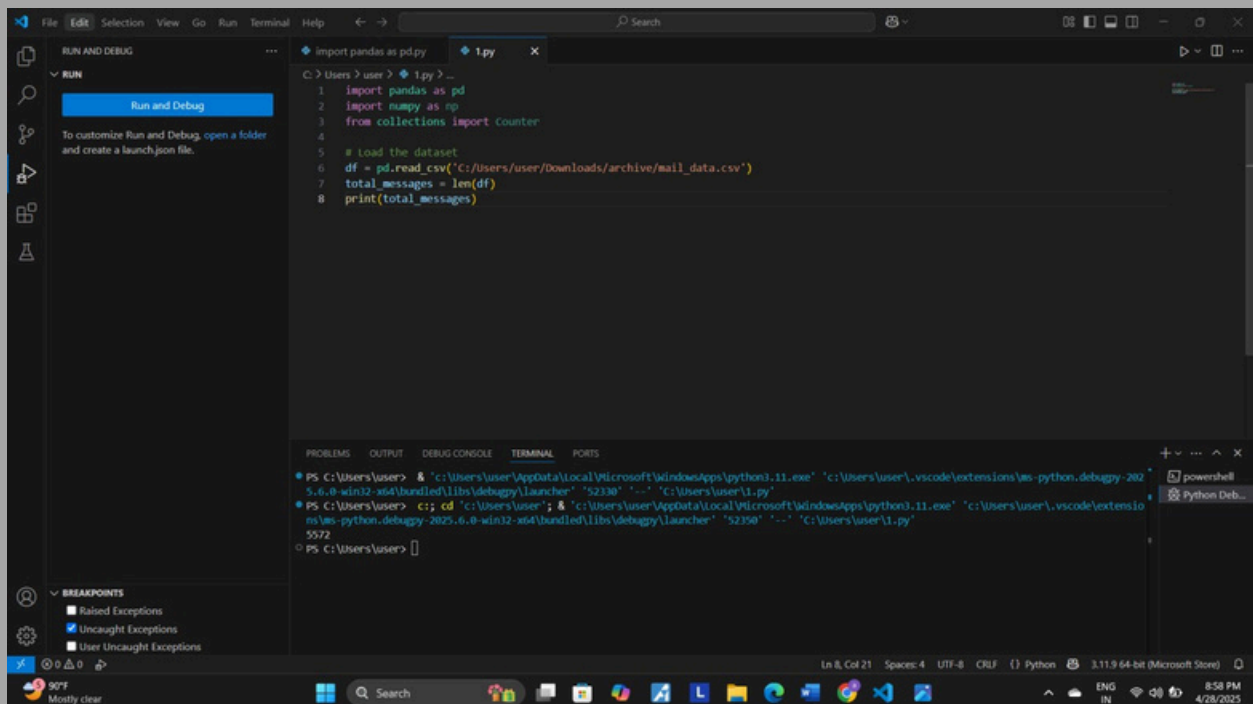


EDS Activity:-

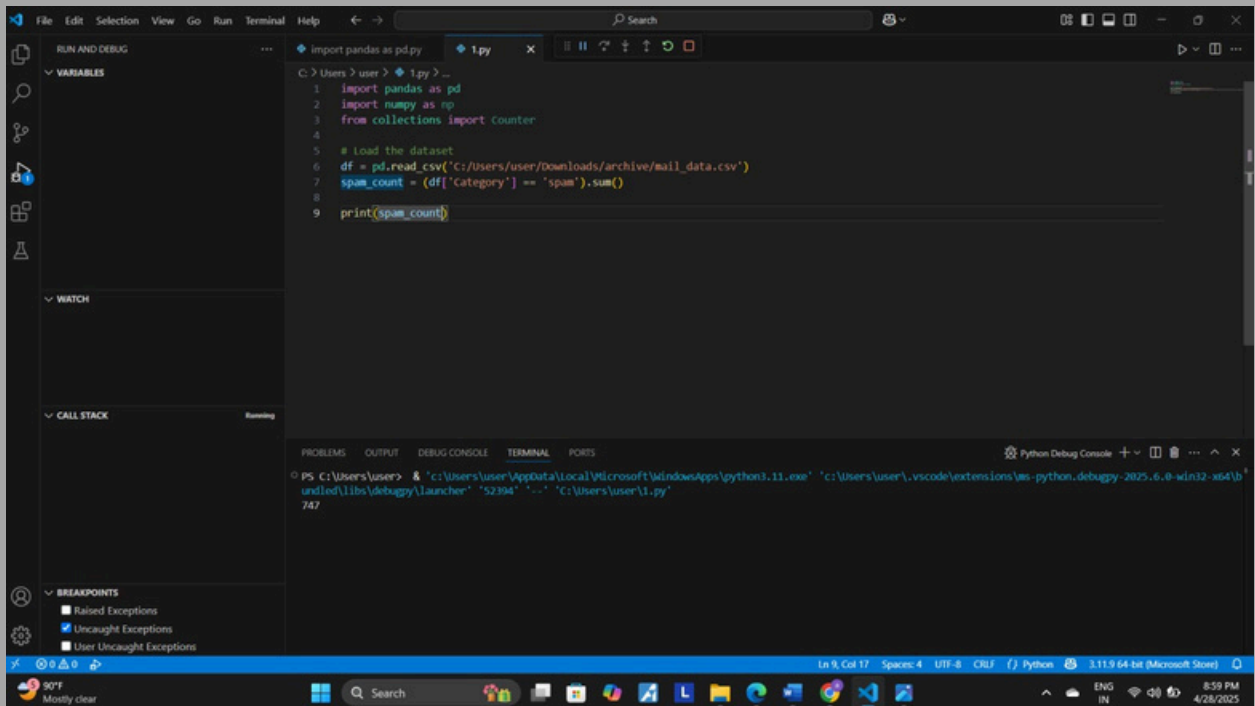
- Name: Kunal Birari
- Roll no.:CS7-27
- PRN:202401080049



1. How many total messages are there?



2. How many messages are spam?

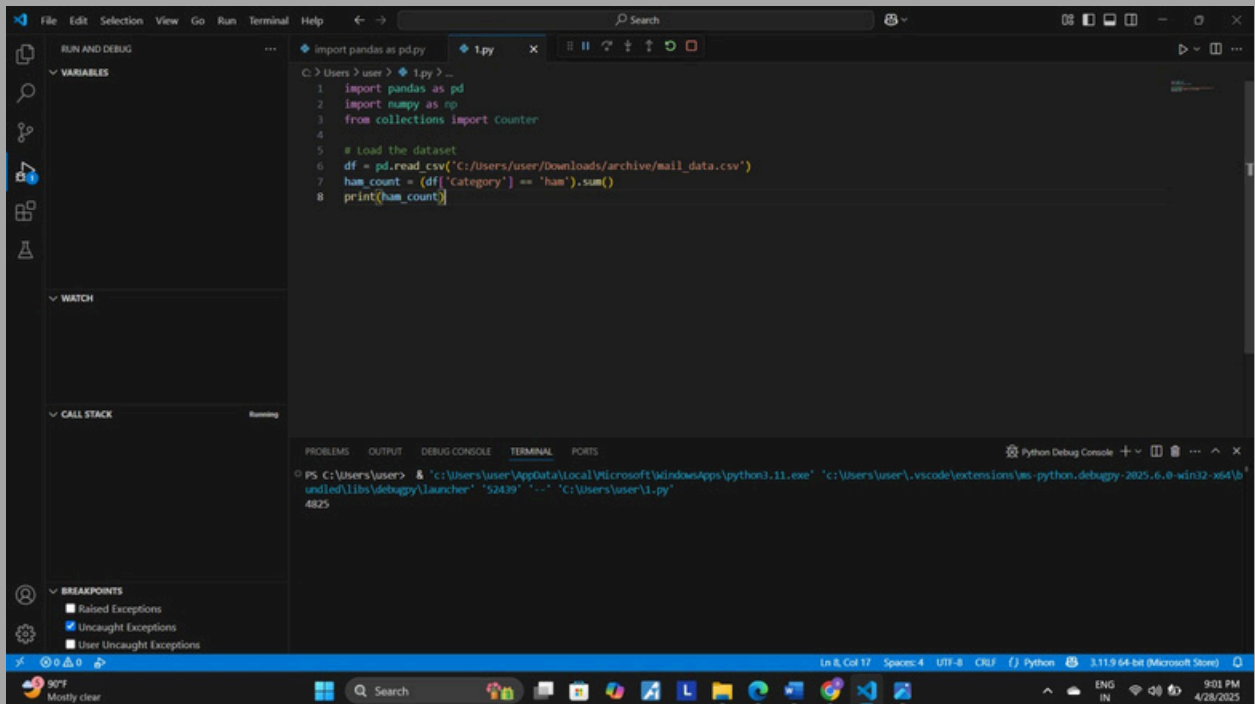


The screenshot shows the Visual Studio Code interface with a Python file named `1.py` open. The script imports `pandas` as `pd`, `numpy` as `np`, and `Counter` from `collections`. It then reads a CSV file located at `C:/Users/user/Downloads/archive/mail_data.csv` into a DataFrame `df`. The script filters for messages where the `category` is `'spam'` and calculates the sum of the `spam_count` column. The output is printed as `spam_count`.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7 spam_count = (df['category'] == 'spam').sum()
8
9 print(spam_count)
```

The terminal window at the bottom shows the command prompt output: `PS C:\Users\user> & 'c:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.11.exe' 'c:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.0-win32-x64\bin\debugpy\launcher' '52394' '-' 'C:\Users\user\1.py'` followed by the output `747`.

3. How many messages are ham?

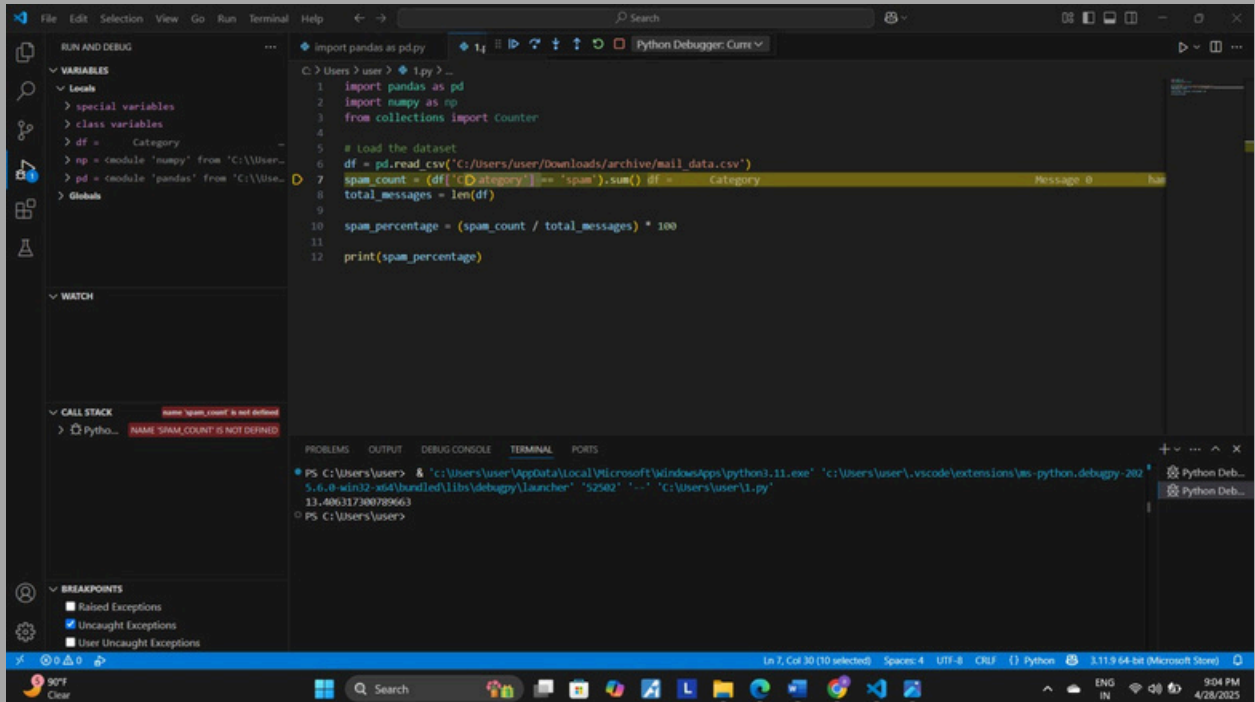


The screenshot shows the Visual Studio Code interface with a Python file named `1.py` open. The script imports `pandas` as `pd`, `numpy` as `np`, and `Counter` from `collections`. It then reads a CSV file located at `C:/Users/user/Downloads/archive/mail_data.csv` into a DataFrame `df`. The script filters for messages where the `category` is `'ham'` and calculates the sum of the `ham_count` column. The output is printed as `ham_count`.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7 ham_count = (df['category'] == 'ham').sum()
8
9 print(ham_count)
```

The terminal window at the bottom shows the command prompt output: `PS C:\Users\user> & 'c:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.11.exe' 'c:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.0-win32-x64\bin\debugpy\launcher' '52439' '-' 'C:\Users\user\1.py'` followed by the output `4825`.

4. What is the percentage of spam messages?



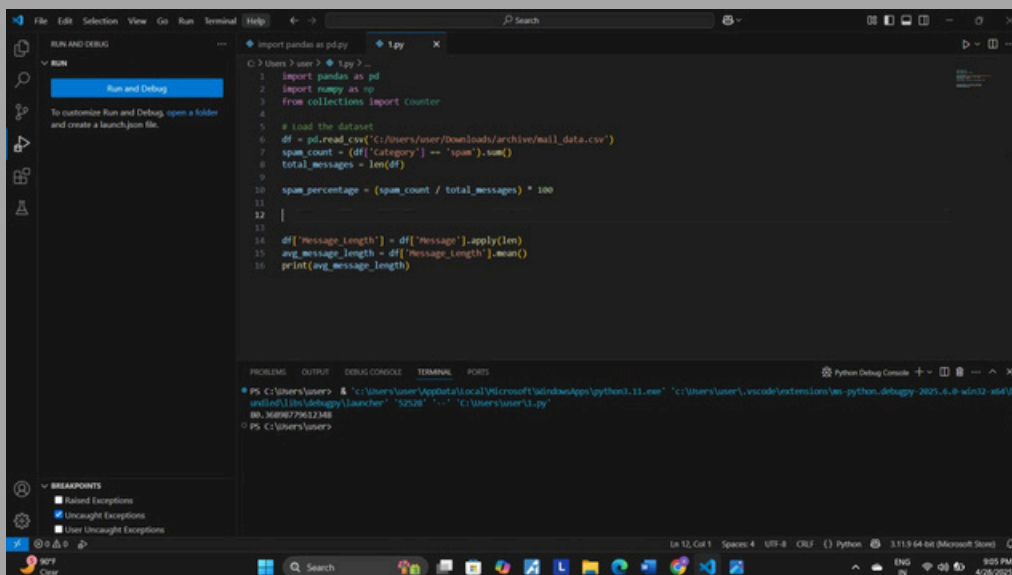
The screenshot shows the Visual Studio Code editor with a Python file named `1.py`. The script imports `pandas` and `numpy`, reads a CSV file from the path `C:/Users/user/Downloads/archive/mail_data.csv`, and calculates the percentage of spam messages. The `Category` column is used to filter for 'spam' messages. The `total_messages` variable is set to the length of the DataFrame. The `spam_percentage` is calculated as $(\text{spam_count} / \text{total_messages}) * 100$ and printed. The left sidebar shows the 'RUN AND DEBUG' panel with 'VARIABLES' and 'WATCH' sections. The 'VARIABLES' section shows `df` as a `Category` object. The 'WATCH' section shows `NAME 'spam_count' IS NOT DEFINED`. The bottom status bar indicates the file is at line 7, column 10, with 10 selected characters, 4 spaces, UTF-8 encoding, and CRLF line endings.

```
import pandas as pd
import numpy as np
from collections import Counter

# Load the dataset
df = pd.read_csv("C:/Users/user/Downloads/archive/mail_data.csv")
spam_count = (df["Category"] == "spam").sum()
total_messages = len(df)

spam_percentage = (spam_count / total_messages) * 100
print(spam_percentage)
```

5. What is the average length of message?



The screenshot shows the Visual Studio Code editor with a Python file named `1.py`. The script imports `pandas` and `numpy`, reads a CSV file from the path `C:/Users/user/Downloads/archive/mail_data.csv`, and calculates the average length of messages. The `Message` column is used to filter for 'spam' messages. The `avg_message_length` is calculated as the mean of the `Message` column. The `print(avg_message_length)` statement is used to output the result. The left sidebar shows the 'RUN AND DEBUG' panel with 'VARIABLES' and 'WATCH' sections. The 'VARIABLES' section shows `df` as a `Category` object. The 'WATCH' section shows `NAME 'spam_count' IS NOT DEFINED`. The bottom status bar indicates the file is at line 12, column 1, with 1 selected character, 4 spaces, UTF-8 encoding, and CRLF line endings.

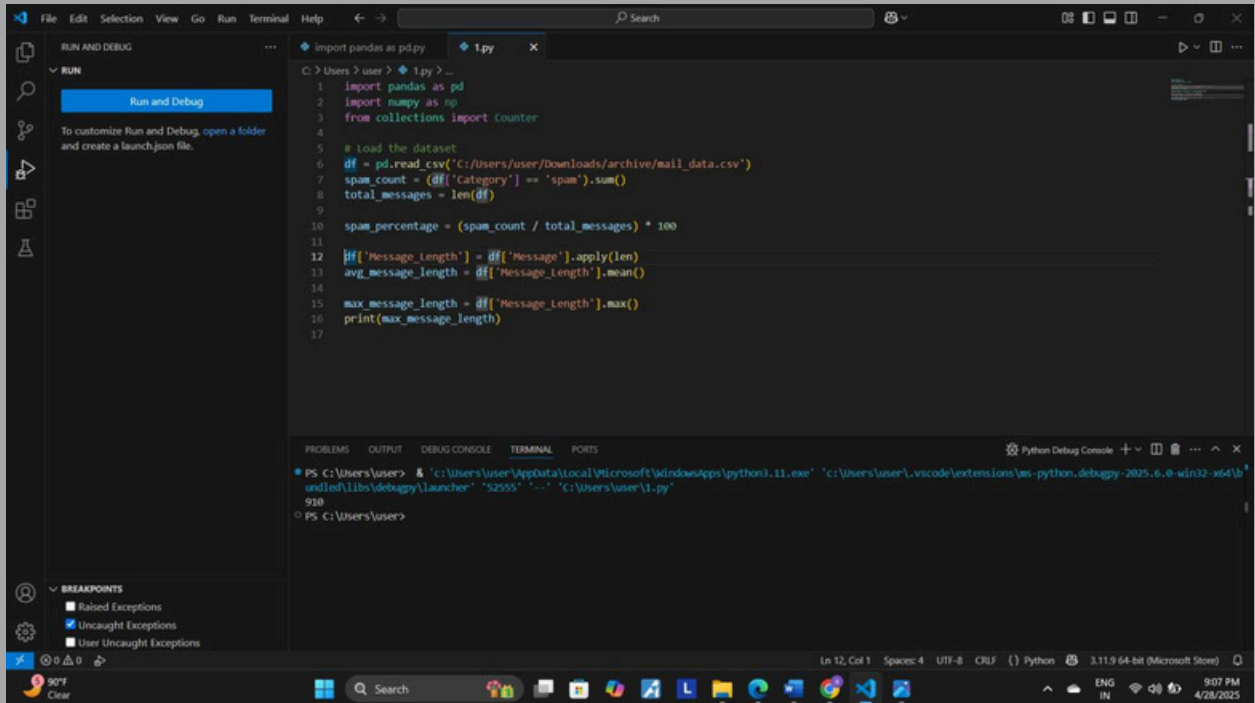
```
import pandas as pd
import numpy as np
from collections import Counter

# Load the dataset
df = pd.read_csv("C:/Users/user/Downloads/archive/mail_data.csv")
spam_count = (df["Category"] == "spam").sum()
total_messages = len(df)

spam_percentage = (spam_count / total_messages) * 100
print(spam_percentage)
```

```
df["Message Length"] = df["Message"].apply(len)
avg_message_length = df["Message Length"].mean()
print(avg_message_length)
```

6. What is the maximum message length?



The screenshot shows a Visual Studio Code window with a Python file named `1.py` open. The script imports `pandas` and `mumpy` (sic), reads a CSV file, and calculates the maximum message length. The terminal output shows the script running successfully.

```
import pandas as pd
import mumpy as mp
from collections import Counter

# Load the dataset
df = pd.read_csv('C:/Users/user/downloads/archive/mail_data.csv')
spam_count = (df['Category'] == 'spam').sum()
total_messages = len(df)

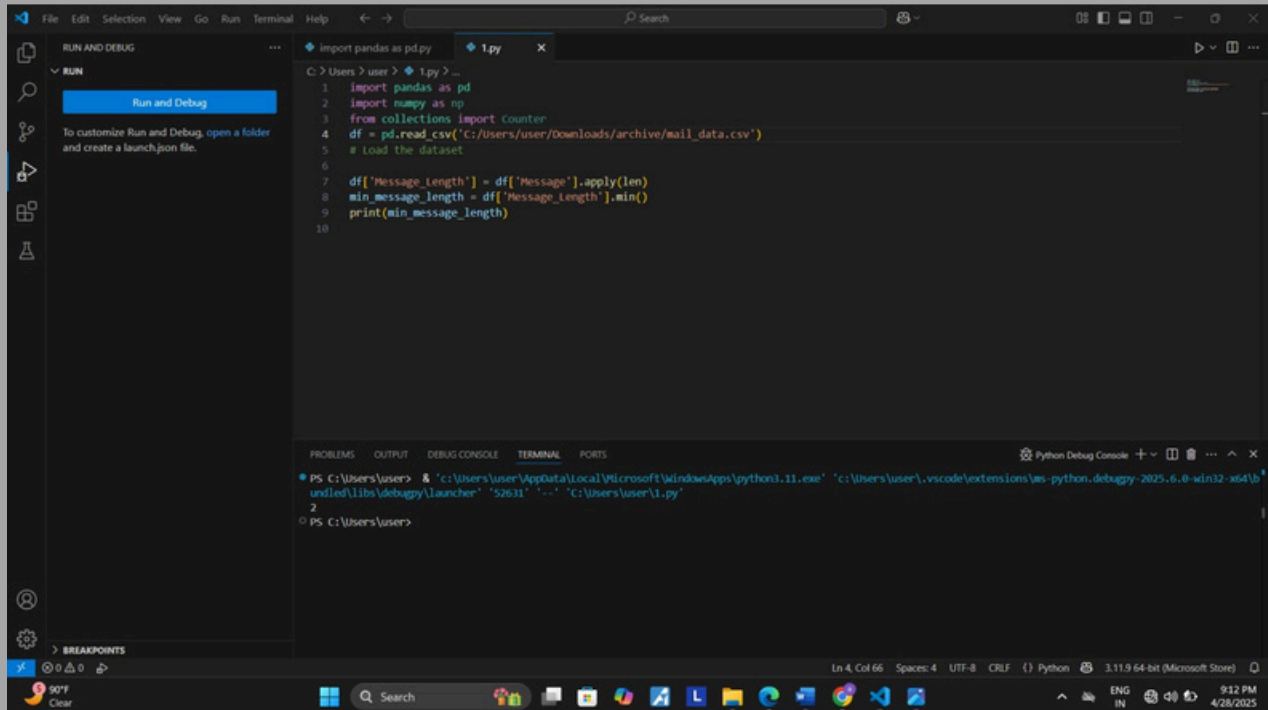
spam_percentage = (spam_count / total_messages) * 100

df['Message_Length'] = df['Message'].apply(len)
avg_message_length = df['Message_Length'].mean()
max_message_length = df['Message_Length'].max()
print(max_message_length)
```

Terminal Output:

```
PS C:\Users\user> & "c:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.11.exe" "c:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.0-win32-x64\bin\unified\libs\debugpy\launcher" "52555" "--" "C:\Users\user\1.py"
910
PS C:\Users\user>
```

7. What is the minimum message length?



The screenshot shows a Visual Studio Code editor window with a Python file named `1.py`. The script imports `pandas` and `numpy`, reads a CSV file, and calculates the minimum message length. The terminal output shows the command used to run the script and the resulting output.

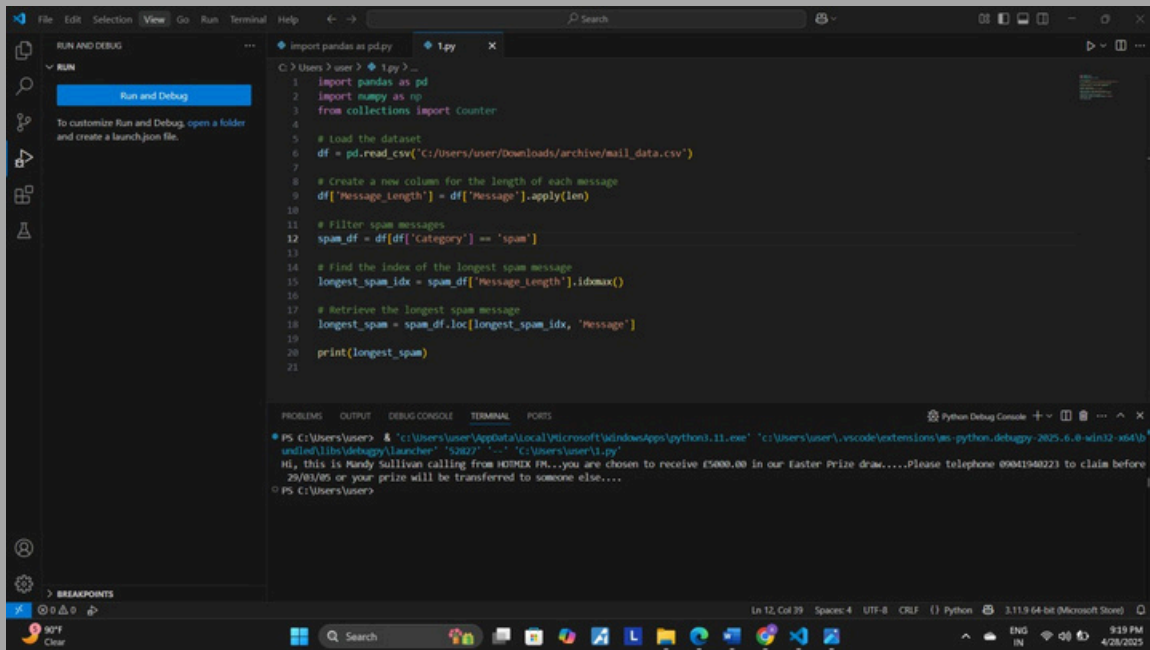
```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
5 # load the dataset
6
7 df['Message_length'] = df['Message'].apply(len)
8 min_message_length = df['Message_length'].min()
9 print(min_message_length)
10
```

Terminal Output:

```
PS C:\Users\user> & "c:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.11.exe" "c:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.0-win32-x64\bin\unified\libs\debugpy\launcher" "52631" "--" "C:\Users\user\1.py"
2
PS C:\Users\user>
```

VS Code status bar: Ln 4, Col 66, Spaces: 4, UTF-8, CRLF, Python, 3.11.9 64-bit (Microsoft Store), 9:12 PM, 4/28/2025.

8. What is the longest spam message?

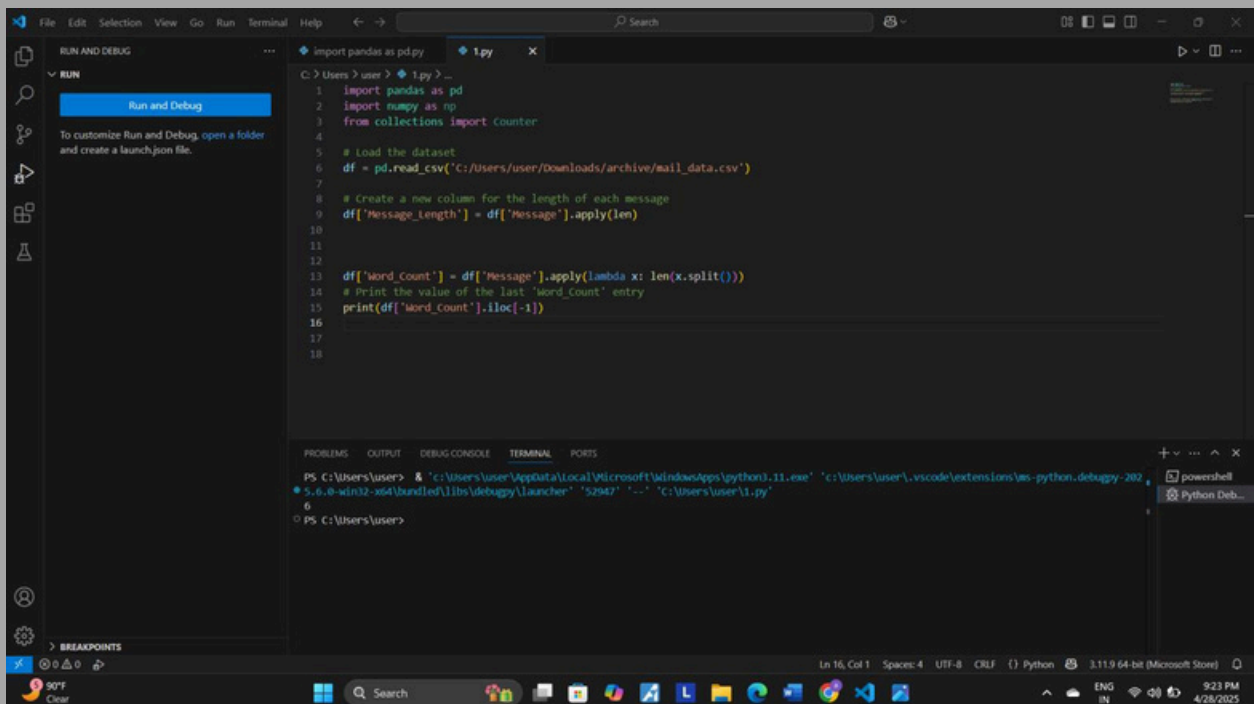


The screenshot shows the Visual Studio Code editor with a Python file named `1.py`. The script imports `pandas`, `numpy`, and `collections`. It reads a CSV file `mail_data.csv` from the user's Downloads folder. A new column `Message_length` is created by applying `len` to the `Message` column. The script filters for spam messages, finds the index of the longest one using `idxmax()`, and prints the message content.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7
8 # Create a new column for the length of each message
9 df['Message_length'] = df['Message'].apply(len)
10
11 # Filter spam messages
12 spam_df = df[df['category'] == 'spam']
13
14 # Find the index of the longest spam message
15 longest_spam_idx = spam_df['Message_length'].idxmax()
16
17 # Retrieve the longest spam message
18 longest_spam = spam_df.loc[longest_spam_idx, 'Message']
19
20 print(longest_spam)
```

The terminal output shows the execution of the script, displaying a long spam message from 'Randy Sullivan'.

10. Create a column with word counts for each message.

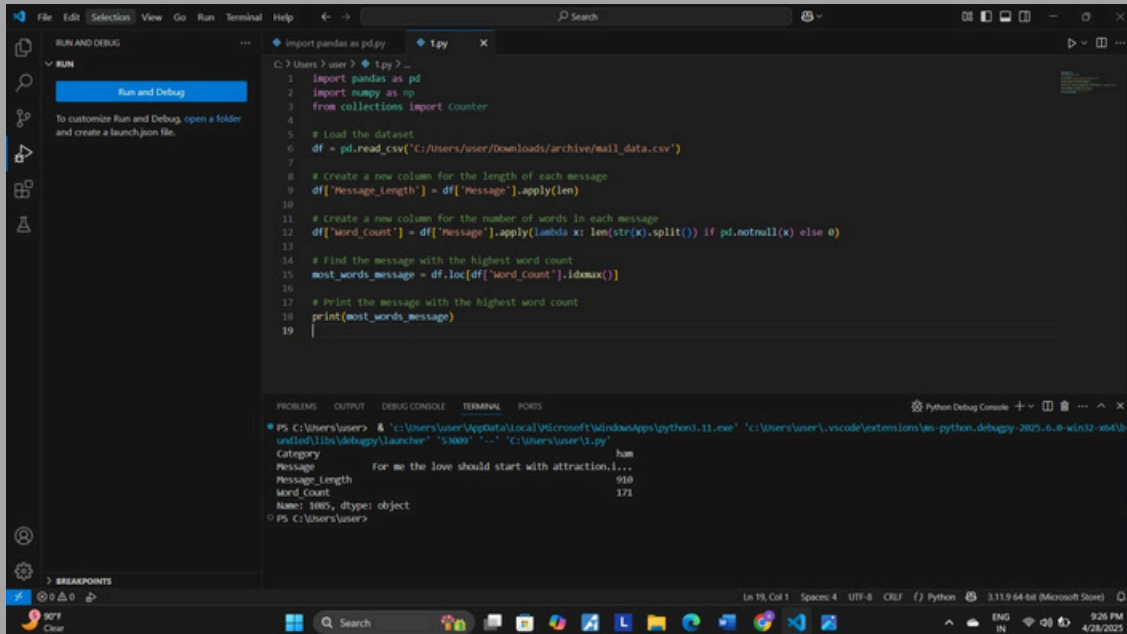


The screenshot shows the Visual Studio Code editor with a Python file named `1.py`. The script imports `pandas`, `numpy`, and `collections`. It reads a CSV file `mail_data.csv` from the user's Downloads folder. A new column `word_count` is created by applying a lambda function that splits the message into words and counts them. The script prints the value of the last entry in the `word_count` column.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7
8 # Create a new column for the length of each message
9 df['Message_length'] = df['Message'].apply(len)
10
11
12
13 df['word_count'] = df['Message'].apply(lambda x: len(x.split()))
14 # Print the value of the last 'word_count' entry
15 print(df['word_count'].iloc[-1])
16
17
18
```

The terminal output shows the execution of the script, displaying the word count of the last message in the dataset.

11. Find the message with the most words.



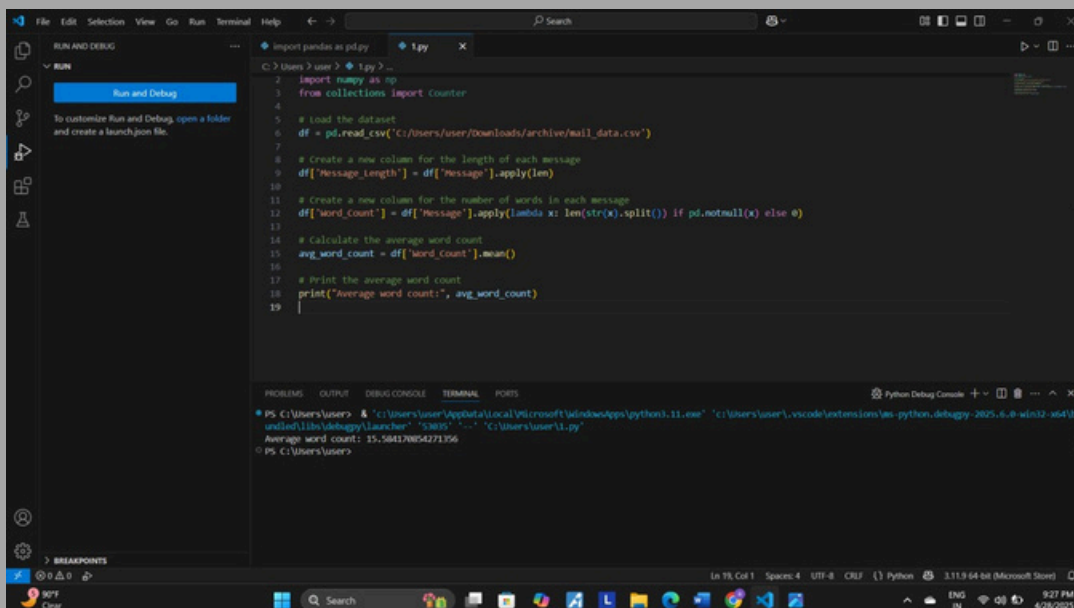
The screenshot shows a VS Code editor with a Python file named `1.py`. The script reads a CSV file, calculates the word count for each message, and prints the message with the highest word count. The output in the terminal shows a message with 910 words.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv("C:/Users/user/Downloads/archive/mail_data.csv")
7
8 # Create a new column for the length of each message
9 df['message_length'] = df['message'].apply(len)
10
11 # Create a new column for the number of words in each message
12 df['word_count'] = df['message'].apply(lambda x: len(str(x).split()) if pd.notnull(x) else 0)
13
14 # Find the message with the highest word count
15 most_words_message = df.loc[df['word_count'].idxmax()]
16
17 # Print the message with the highest word count
18 print(most_words_message)
19
```

Terminal Output:

```
PS C:\Users\user> & "C:/Users/user/AppData/Local/Microsoft/WindowsApps/python.11.exe" "C:/Users/user/.vscode/extensions/ms-python.debugpy-2025.6.0-win32-x64/lib
union/lib/debugpy/launcher" "53009" "..." "C:/Users/user/1.py"
Category: hum
Message: For me the love should start with attraction....
message_length: 910
word_count: 171
Name: 3885, dtype: object
PS C:\Users\user>
```

12. What is the average word count per message?



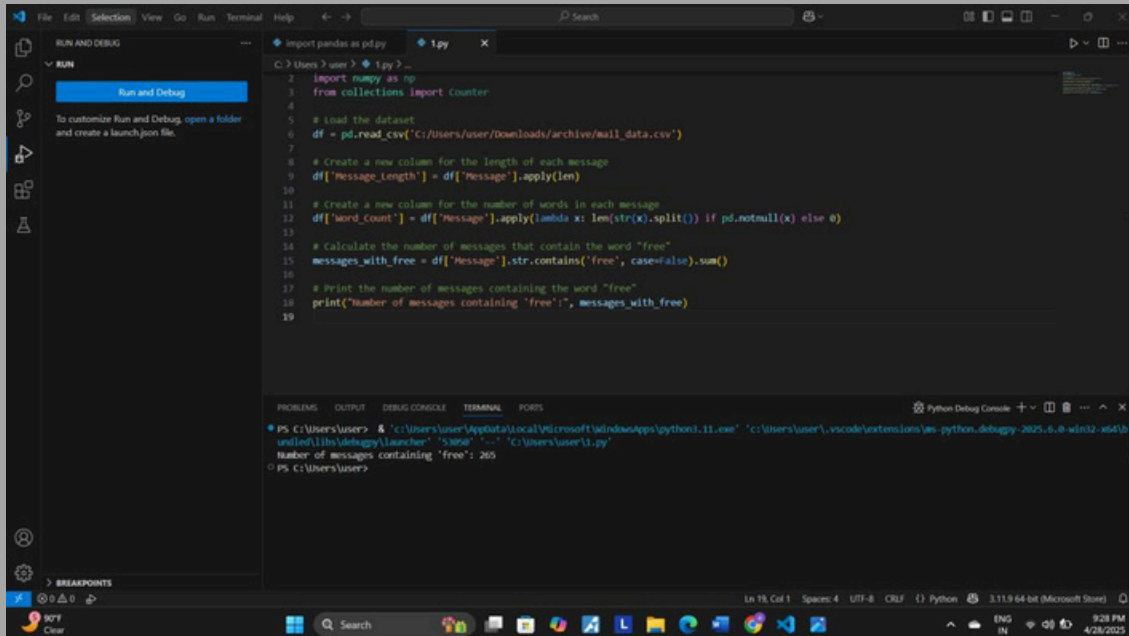
The screenshot shows a VS Code editor with a Python file named `1.py`. The script reads a CSV file, calculates the word count for each message, and prints the average word count. The output in the terminal shows an average word count of 15.5641780427136.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv("C:/Users/user/Downloads/archive/mail_data.csv")
7
8 # Create a new column for the length of each message
9 df['message_length'] = df['message'].apply(len)
10
11 # Create a new column for the number of words in each message
12 df['word_count'] = df['message'].apply(lambda x: len(str(x).split()) if pd.notnull(x) else 0)
13
14 # Calculate the average word count
15 avg_word_count = df['word_count'].mean()
16
17 # Print the average word count
18 print("Average word count:", avg_word_count)
19
```

Terminal Output:

```
PS C:\Users\user> & "C:/Users/user/AppData/Local/Microsoft/WindowsApps/python.11.exe" "C:/Users/user/.vscode/extensions/ms-python.debugpy-2025.6.0-win32-x64/lib
union/lib/debugpy/launcher" "53009" "..." "C:/Users/user/1.py"
Average word count: 15.5641780427136
PS C:\Users\user>
```


13. Find how many messages contain the word "free".



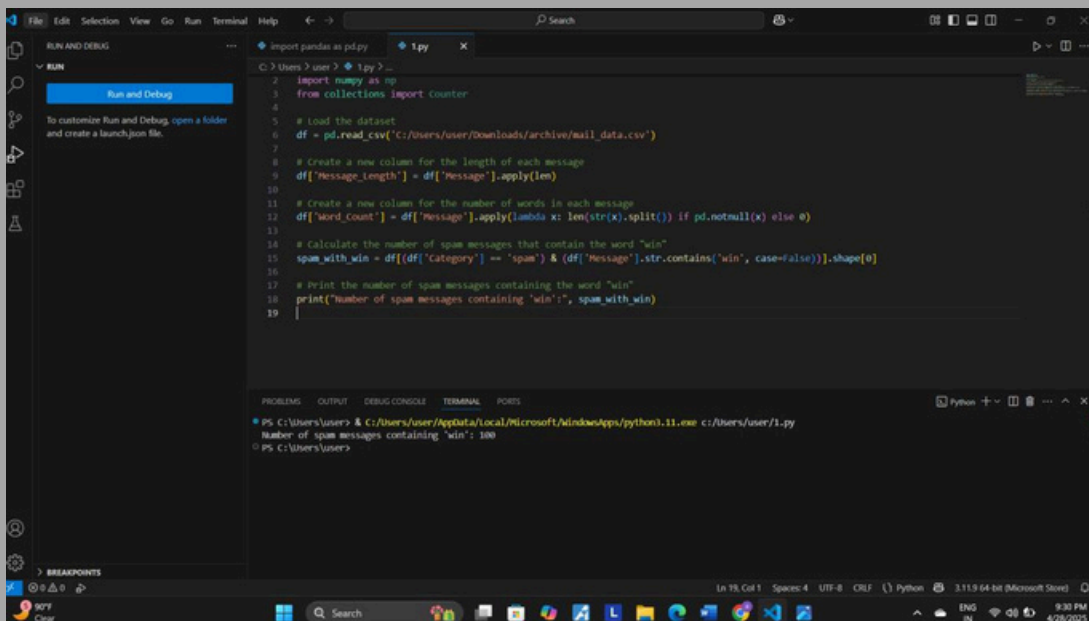
The screenshot shows a Python IDE with a file named '1.py'. The code reads a CSV file, creates a 'word_count' column, and counts messages containing the word 'free'. The terminal output shows the result is 265.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7
8 # Create a new column for the length of each message
9 df['message_length'] = df['Message'].apply(len)
10
11 # Create a new column for the number of words in each message
12 df['word_count'] = df['Message'].apply(lambda x: len(str(x).split()) if pd.notnull(x) else 0)
13
14 # Calculate the number of messages that contain the word "free"
15 messages_with_free = df['Message'].str.contains('free', case=False).sum()
16
17 # Print the number of messages containing the word "free"
18 print("Number of messages containing 'free':", messages_with_free)
19
```

Python Debug Console output:

```
PS C:\Users\user> & 'C:\Users\user\AppData\Local\Microsoft\WindowsApps\python.11.exe' 'C:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.8-win32-x64\bin\python\lib\debugpy\launcher' "Spase" "-." "C:\Users\user\1.py"
Number of messages containing 'free': 265
PS C:\Users\user>
```

14. Find how many spam messages contain the word "win".



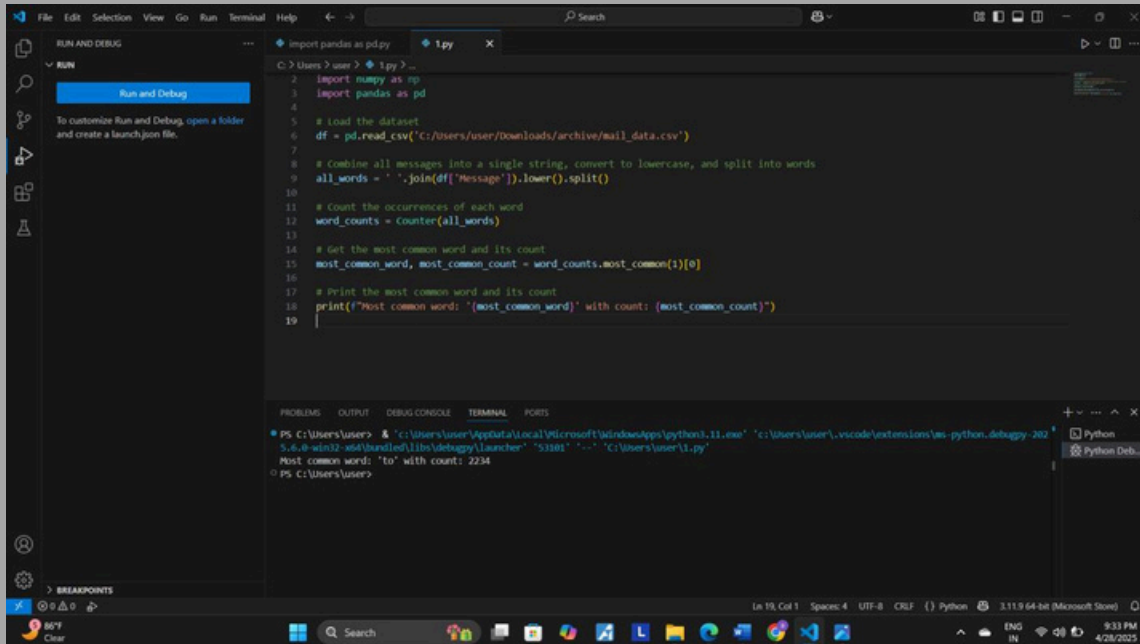
The screenshot shows a Python IDE with a file named '1.py'. The code reads a CSV file, creates a 'word_count' column, and counts spam messages containing the word 'win'. The terminal output shows the result is 100.

```
1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4
5 # Load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7
8 # Create a new column for the length of each message
9 df['message_length'] = df['Message'].apply(len)
10
11 # Create a new column for the number of words in each message
12 df['word_count'] = df['Message'].apply(lambda x: len(str(x).split()) if pd.notnull(x) else 0)
13
14 # Calculate the number of spam messages that contain the word "win"
15 spam_with_win = df[(df['Category'] == 'spam') & (df['Message'].str.contains('win', case=False))].shape[0]
16
17 # Print the number of spam messages containing the word "win"
18 print("Number of spam messages containing 'win':", spam_with_win)
19
```

Python Debug Console output:

```
PS C:\Users\user> & 'C:\Users\user\AppData\Local\Microsoft\WindowsApps\python.11.exe' 'C:\Users\user\1.py'
Number of spam messages containing 'win': 100
PS C:\Users\user>
```

15. What is the most common word in the dataset?



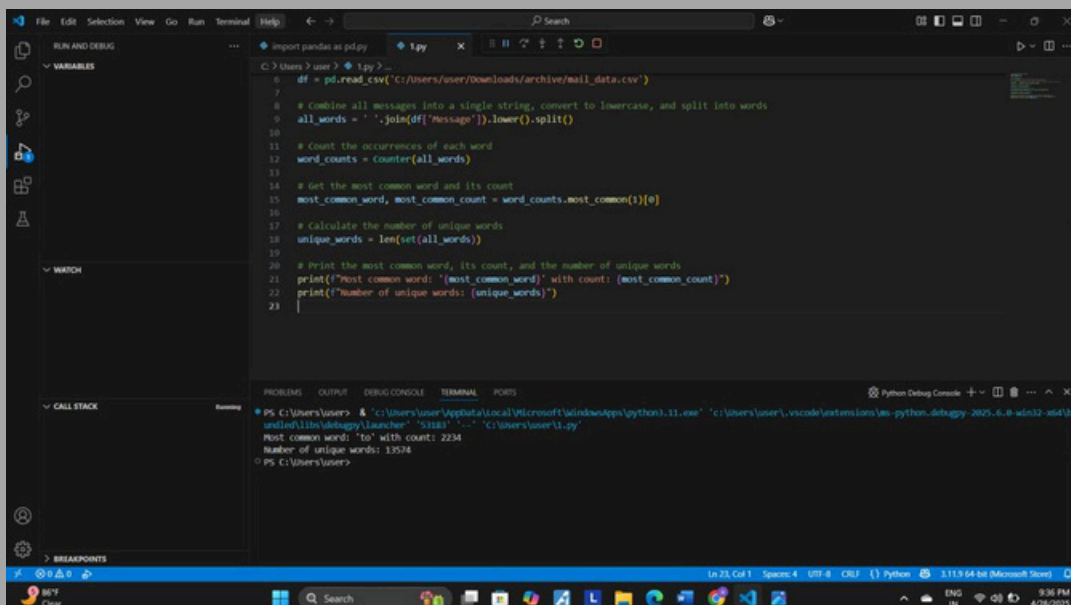
The screenshot shows a VS Code editor with a Python script in a file named `1.py`. The script imports `pandas`, `numpy`, and `pd`. It reads a CSV file `archive/mail_data.csv` into a DataFrame `df`. The script then combines all messages into a single string, converts it to lowercase, and splits it into words. It counts the occurrences of each word using `Counter` and finds the most common word and its count. The output is printed as `Most common word: 'to' with count: 2234`. The terminal at the bottom shows the execution of the script, confirming the output.

```
1 import pandas as pd
2 import numpy as np
3 import pandas as pd
4
5 # load the dataset
6 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
7
8 # Combine all messages into a single string, convert to lowercase, and split into words
9 all_words = ' '.join(df['Message']).lower().split()
10
11 # Count the occurrences of each word
12 word_counts = Counter(all_words)
13
14 # Get the most common word and its count
15 most_common_word, most_common_count = word_counts.most_common(1)[0]
16
17 # Print the most common word and its count
18 print(f"Most common word: '{most_common_word}' with count: {most_common_count}")
19
```

Terminal output:

```
PS C:\Users\user> & "C:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.11.exe" "C:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.0-win32-x64\lib\debugpy\launcher" "51181" "-c" "C:\Users\user\1.py"
Most common word: 'to' with count: 2234
PS C:\Users\user>
```

16. How many unique words are there across all messages?



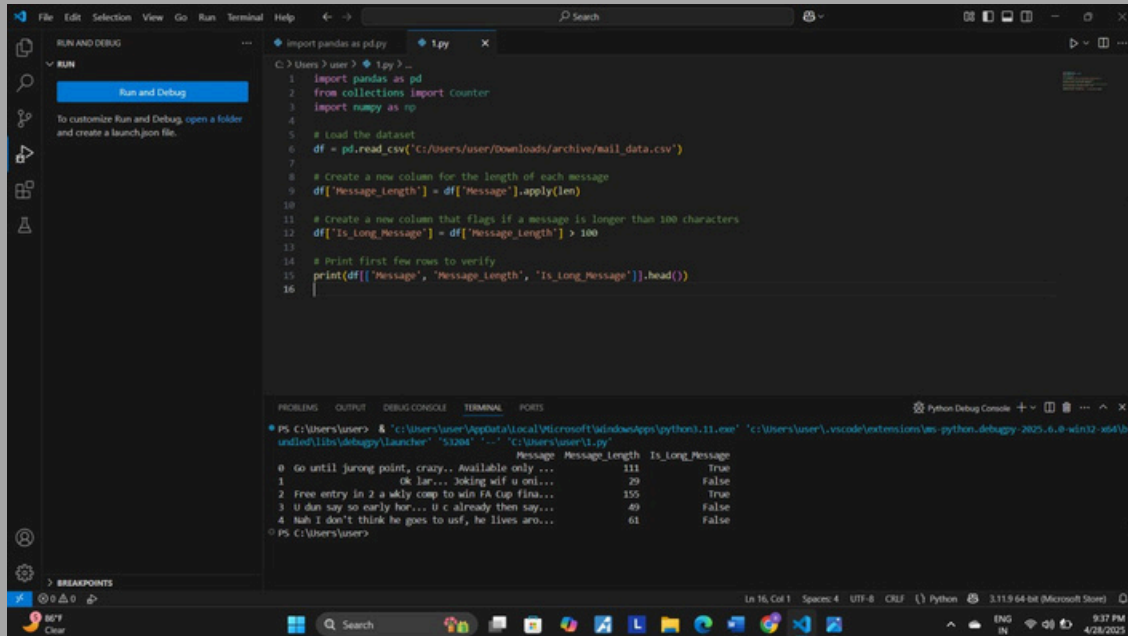
The screenshot shows a VS Code editor with a Python script in a file named `1.py`. The script imports `pandas`, `numpy`, and `pd`. It reads a CSV file `archive/mail_data.csv` into a DataFrame `df`. The script then combines all messages into a single string, converts it to lowercase, and splits it into words. It counts the occurrences of each word using `Counter` and finds the most common word and its count. It also calculates the number of unique words using `len(set(all_words))`. The output is printed as `Most common word: 'to' with count: 2234` and `Number of unique words: 13574`. The terminal at the bottom shows the execution of the script, confirming the output.

```
1 df = pd.read_csv('C:/Users/user/Downloads/archive/mail_data.csv')
2
3 # Combine all messages into a single string, convert to lowercase, and split into words
4 all_words = ' '.join(df['Message']).lower().split()
5
6 # Count the occurrences of each word
7 word_counts = Counter(all_words)
8
9 # Get the most common word and its count
10 most_common_word, most_common_count = word_counts.most_common(1)[0]
11
12 # Calculate the number of unique words
13 unique_words = len(set(all_words))
14
15 # Print the most common word, its count, and the number of unique words
16 print(f"Most common word: '{most_common_word}' with count: {most_common_count}")
17 print(f"Number of unique words: {unique_words}")
18
```

Terminal output:

```
PS C:\Users\user> & "C:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.11.exe" "C:\Users\user\.vscode\extensions\ms-python.debugpy-2025.6.0-win32-x64\lib\debugpy\launcher" "51181" "-c" "C:\Users\user\1.py"
Most common word: 'to' with count: 2234
Number of unique words: 13574
PS C:\Users\user>
```

17. Create a boolean column: Is the message length > 100?

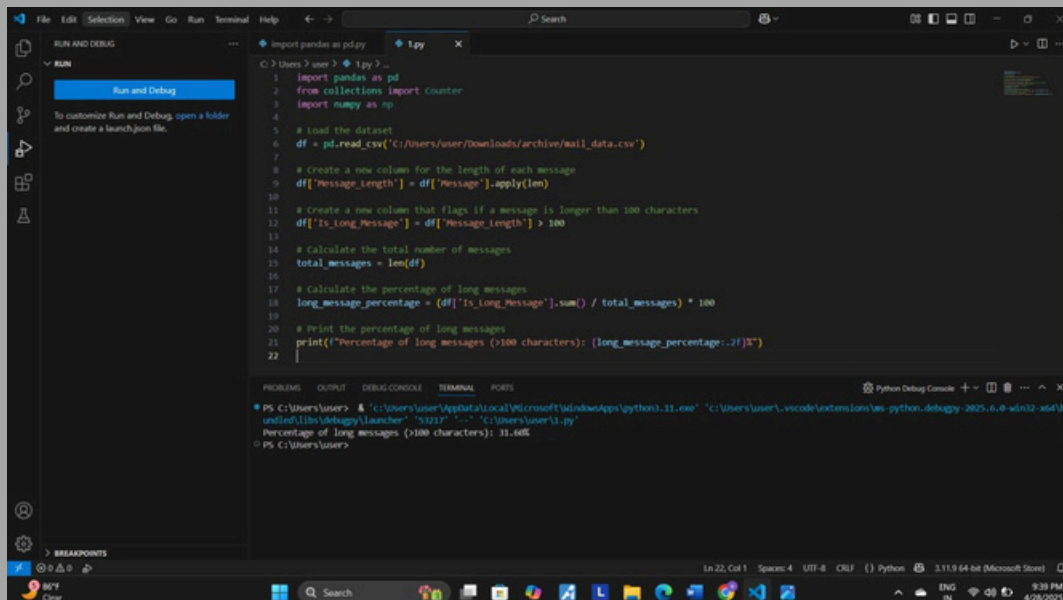


The screenshot shows a VS Code editor with a Python file named `1.py`. The script reads a CSV file, creates a new column for message length, and a boolean column for messages longer than 100 characters. The terminal output shows the first few rows of the resulting DataFrame.

```
1 import pandas as pd
2 from collections import Counter
3 import numpy as np
4
5 # Load the dataset
6 df = pd.read_csv("C:/Users/user/Downloads/archive/mail_data.csv")
7
8 # Create a new column for the length of each message
9 df['Message_length'] = df['Message'].apply(len)
10
11 # Create a new column that flags if a message is longer than 100 characters
12 df['Is_long_Message'] = df['Message_length'] > 100
13
14 # Print first few rows to verify
15 print(df[['Message', 'Message_length', 'Is_long_Message']].head())
16
```

	Message	Message_length	Is_long_Message
0	Go until Jurong point, crazy.. Available only ...	111	True
1	Ok lar... Joking wif u oni...	29	False
2	Free entry in 2 a wky comp to win FA Cup fina...	155	True
3	U dun say so early hor... U c already then say...	49	False
4	Nah I don't think he goes to usf, he lives ar...	61	False

18. Find the percentage of long messages (>100 characters).

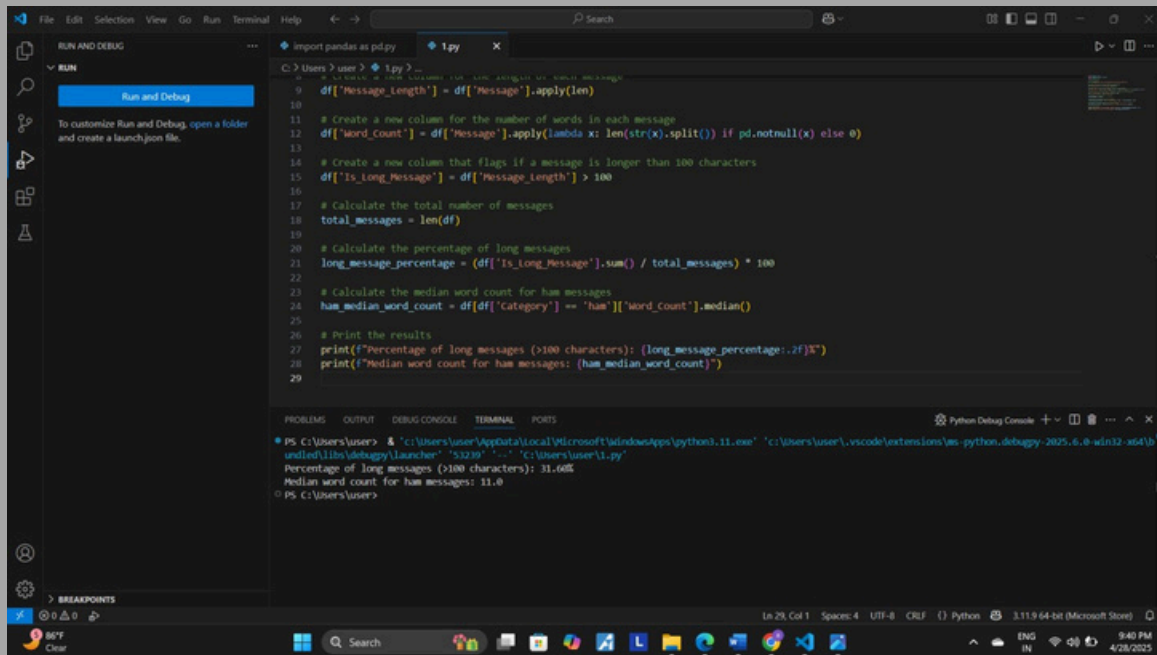


The screenshot shows a VS Code editor with a Python file named `1.py`. The script calculates the total number of messages and the percentage of messages longer than 100 characters. The terminal output shows the final result.

```
1 import pandas as pd
2 from collections import Counter
3 import numpy as np
4
5 # Load the dataset
6 df = pd.read_csv("C:/Users/user/Downloads/archive/mail_data.csv")
7
8 # Create a new column for the length of each message
9 df['Message_length'] = df['Message'].apply(len)
10
11 # Create a new column that flags if a message is longer than 100 characters
12 df['Is_long_Message'] = df['Message_length'] > 100
13
14 # Calculate the total number of messages
15 total_messages = len(df)
16
17 # Calculate the percentage of long messages
18 long_message_percentage = (df['Is_long_Message'].sum() / total_messages) * 100
19
20 # Print the percentage of long messages
21 print(("Percentage of long messages (>100 characters): {:.4f}%".format(long_message_percentage)))
22
```

Percentage of long messages (>100 characters): 31.66%

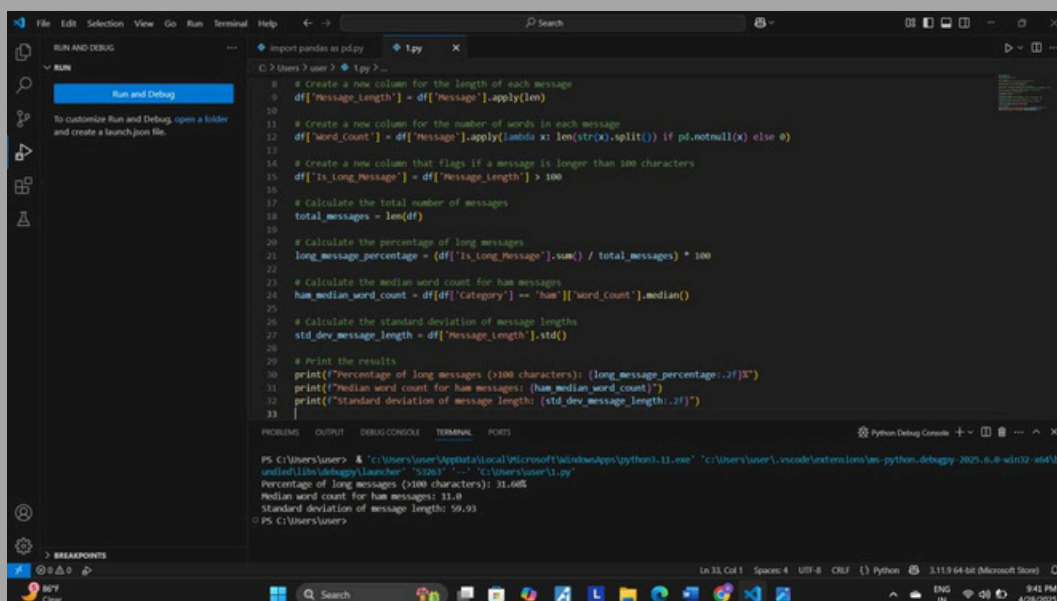
19. What is the median word count of ham messages?



The screenshot shows a VS Code editor with a Python file named `1.py`. The script uses pandas to analyze a dataset of messages. It calculates the median word count for ham messages and prints the results. The terminal output shows the following results:

```
Percentage of long messages (>100 characters): 31.60%
Median word count for ham messages: 11.0
```

20. Find the standard deviation of message lengths.



The screenshot shows a VS Code editor with a Python file named `1.py`. The script uses pandas to analyze a dataset of messages. It calculates the standard deviation of message lengths and prints the results. The terminal output shows the following results:

```
Percentage of long messages (>100 characters): 31.60%
Median word count for ham messages: 11.0
Standard deviation of message length: 59.93
```