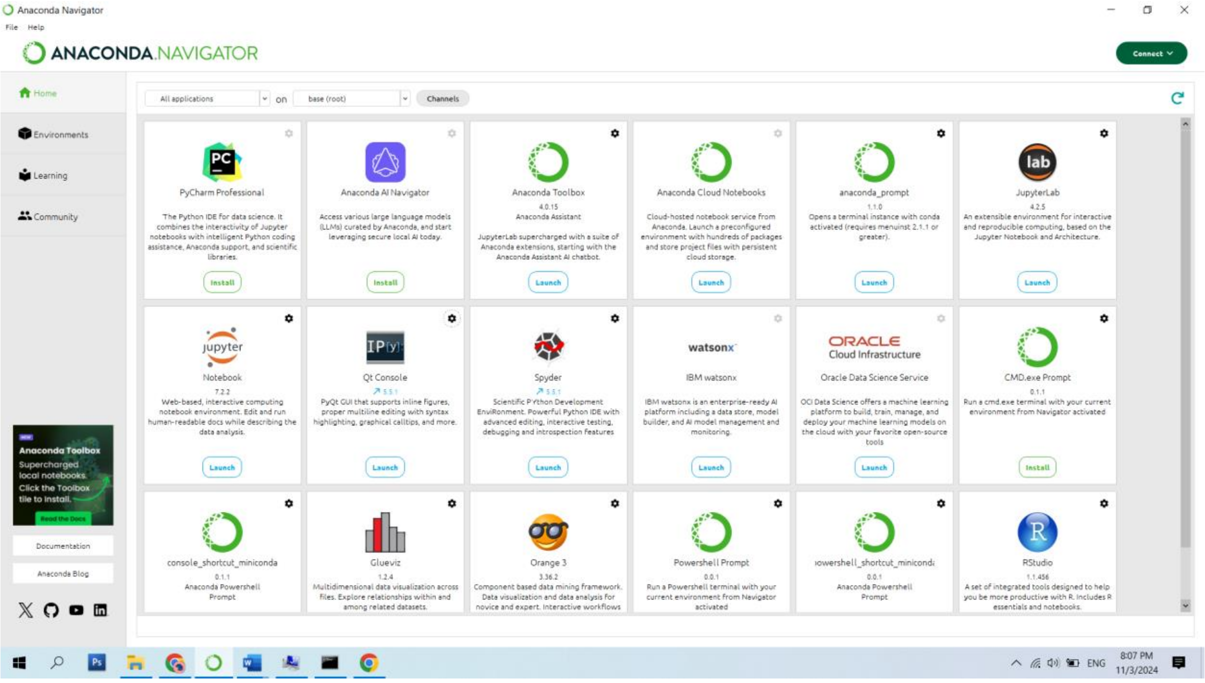
CLOUD ASSIGNMENT - 4

Big Data with PySpark using Anaconda & Jupyter notebook

1)



PyCharm Professional
The Python IDE for data science. It combines the interactivity of Jupyter notebooks with intelligent Python coding assistance, Anaconda support, and scientific libraries.
Install

Anaconda AI Navigator
Access various large language models (LLMs) curated by Anaconda, and start leveraging secure local AI today.
Install

Anaconda Toolbox
4.0.15
Anaconda Assistant
JupyterLab supercharged with a suite of Anaconda extensions, starting with the Anaconda Assistant AI chatbot.
Launch

Anaconda Cloud Notebooks
Cloud-hosted notebook service from Anaconda. Launch a preconfigured environment with hundreds of packages and store project files with persistent cloud storage.
Launch

anaconda_prompt
1.1.0
Opens a terminal instance with conda activated (requires menuinst 2.1.1 or greater).
Launch

JupyterLab
4.2.5
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

Notebook
7.2.2
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

Qt Console
↗ 5.5.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

Spyder
↗ 5.5.1
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch

IBM watsonx
IBM watsonx is an enterprise-ready AI platform including a data store, model builder, and AI model management and monitoring.
Launch

Oracle Data Science Service
OCI Data Science offers a machine learning platform to build, train, manage, and deploy your machine learning models on the cloud with your favorite open-source tools
Launch

CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated
Install

console_shortcut_miniconda
0.1.1
Anaconda Powershell Prompt

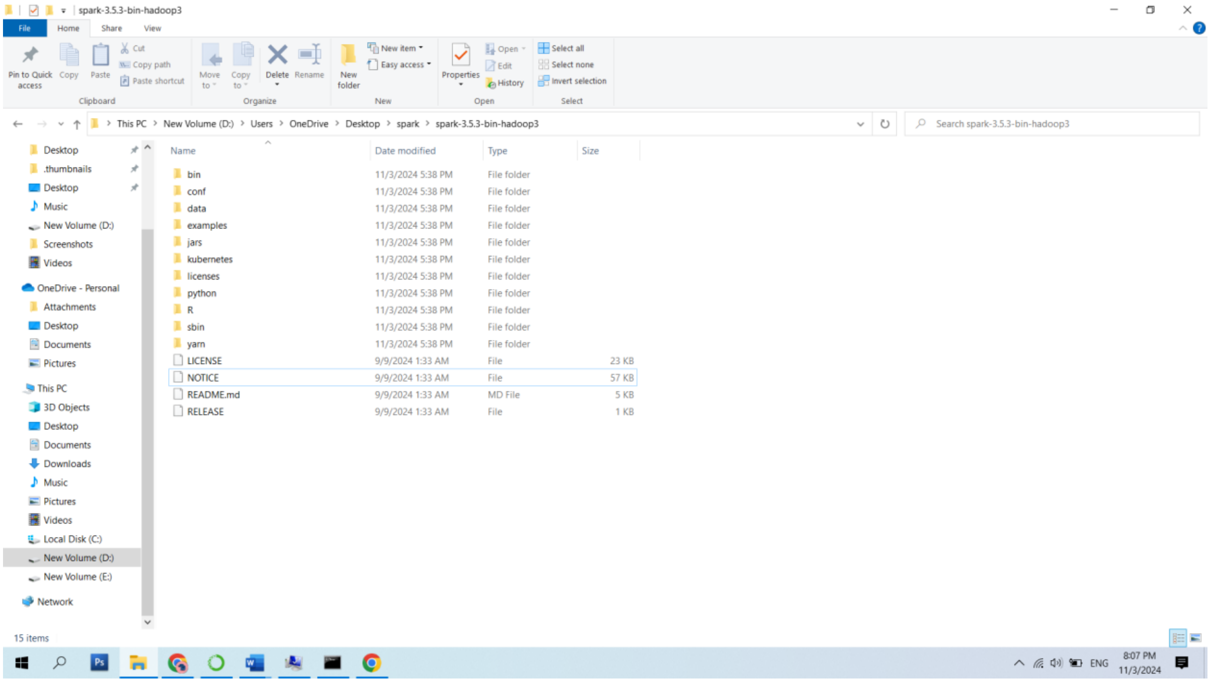
Glueviz
1.2.4
Multidimensional data visualization across files. Explore relationships within and among related datasets.

Orange 3
3.36.2
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows

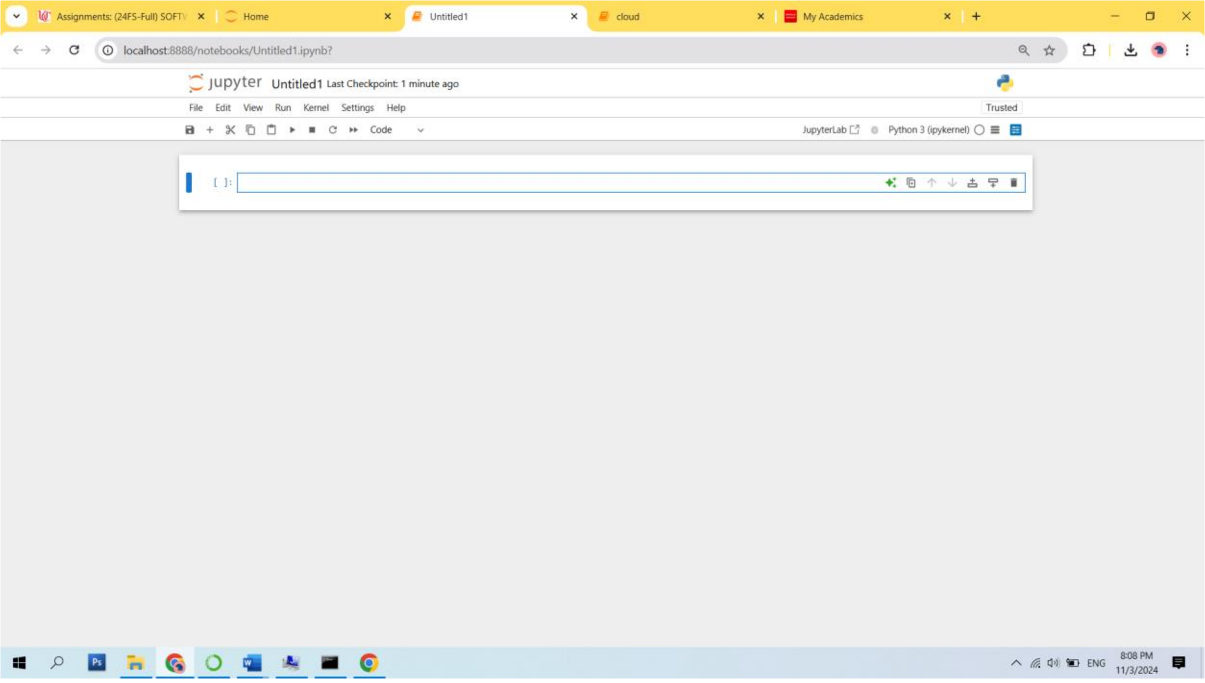
Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated

ıowershell_shortcut_miniconda
0.0.1
Anaconda Powershell Prompt

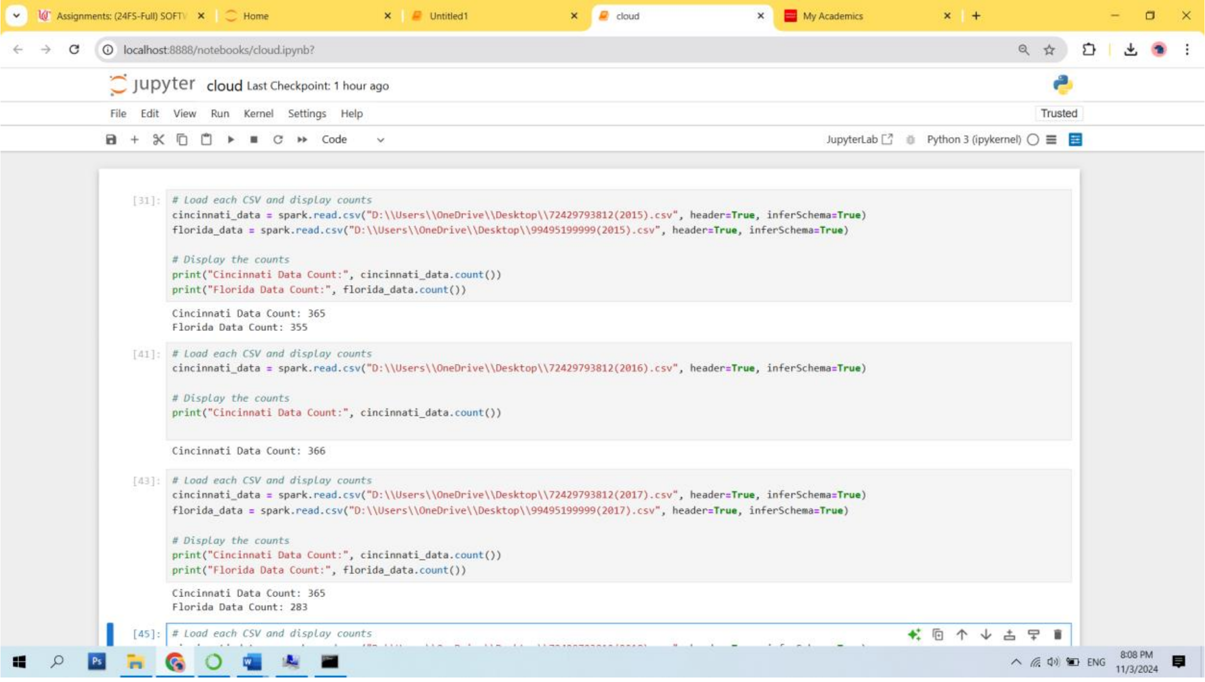
RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.



Name	Date modified	Type	Size
bin	11/3/2024 5:38 PM	File folder	
conf	11/3/2024 5:38 PM	File folder	
data	11/3/2024 5:38 PM	File folder	
examples	11/3/2024 5:38 PM	File folder	
jars	11/3/2024 5:38 PM	File folder	
kubernetes	11/3/2024 5:38 PM	File folder	
licenses	11/3/2024 5:38 PM	File folder	
python	11/3/2024 5:38 PM	File folder	
R	11/3/2024 5:38 PM	File folder	
sbin	11/3/2024 5:38 PM	File folder	
yarn	11/3/2024 5:38 PM	File folder	
LICENSE	9/9/2024 1:33 AM	File	23 KB
NOTICE	9/9/2024 1:33 AM	File	57 KB
README.md	9/9/2024 1:33 AM	MD File	5 KB
RELEASE	9/9/2024 1:33 AM	File	1 KB


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/Untitled1.ipynb?
jupyter Untitled1 Last Checkpoint: 1 minute ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[ ]:
8:08 PM
11/3/2024
ENG

2)

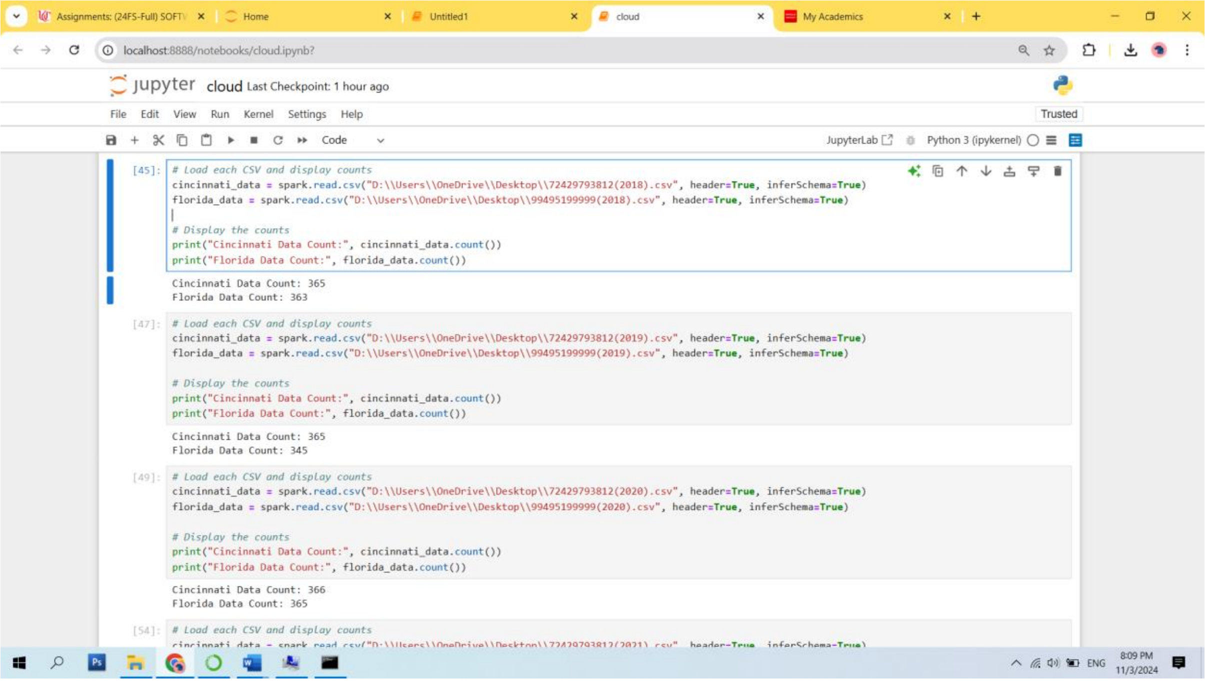

Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
8:08 PM
11/3/2024
ENG

```
[31]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2015).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2015).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 355
```

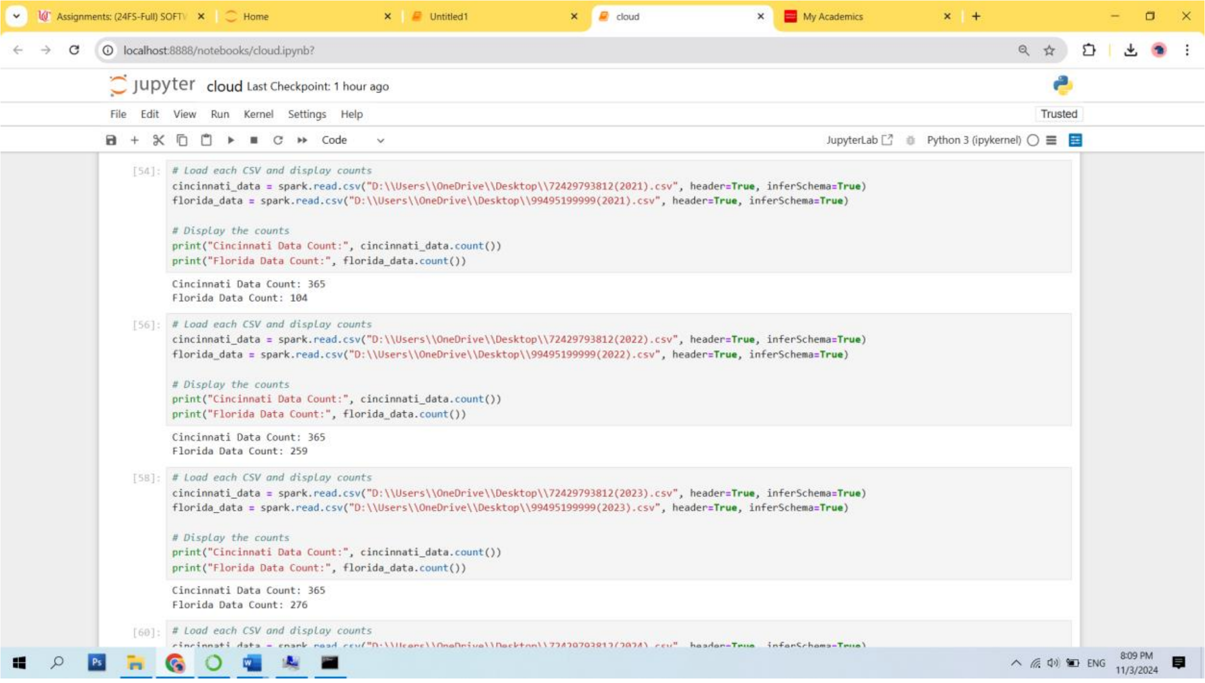
```
[41]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2016).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())

Cincinnati Data Count: 366
```

```
[43]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2017).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2017).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 283
```

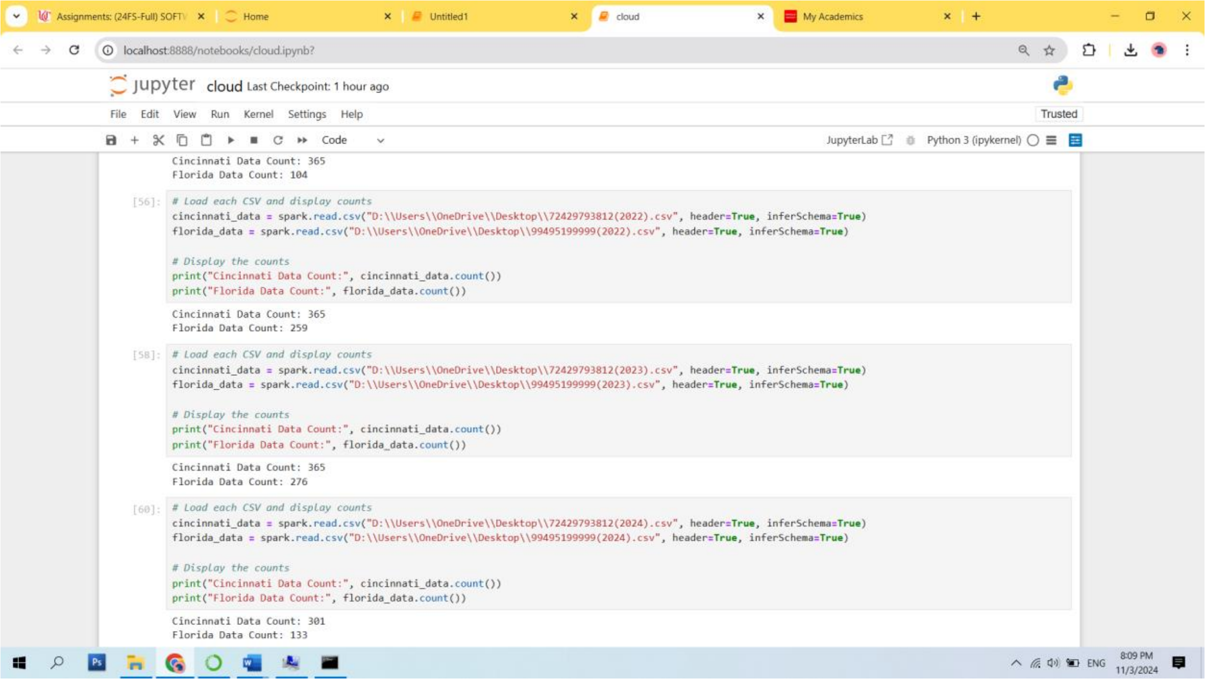
```
[45]: # Load each CSV and display counts
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:09 PM
11/3/2024

```
[45]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2018).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2018).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 363

[47]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2019).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2019).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 345

[49]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2020).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2020).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 366
Florida Data Count: 365

[54]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2021).csv", header=True, inferSchema=True)
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:09 PM
11/3/2024

```
[54]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2021).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2021).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 104

[56]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2022).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2022).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 259

[58]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2023).csv", header=True, inferSchema=True)
florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2023).csv", header=True, inferSchema=True)

# Display the counts
print("Cincinnati Data Count:", cincinnati_data.count())
print("Florida Data Count:", florida_data.count())

Cincinnati Data Count: 365
Florida Data Count: 276

[60]: # Load each CSV and display counts
cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2024).csv", header=True, inferSchema=True)
```



Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
8:09 PM
11/3/2024
ENG

```
Cincinnati Data Count: 365
Florida Data Count: 104
```

```
[56]: # Load each CSV and display counts
      cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2022).csv", header=True, inferSchema=True)
      florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2022).csv", header=True, inferSchema=True)

      # Display the counts
      print("Cincinnati Data Count:", cincinnati_data.count())
      print("Florida Data Count:", florida_data.count())
```

```
Cincinnati Data Count: 365
Florida Data Count: 259
```

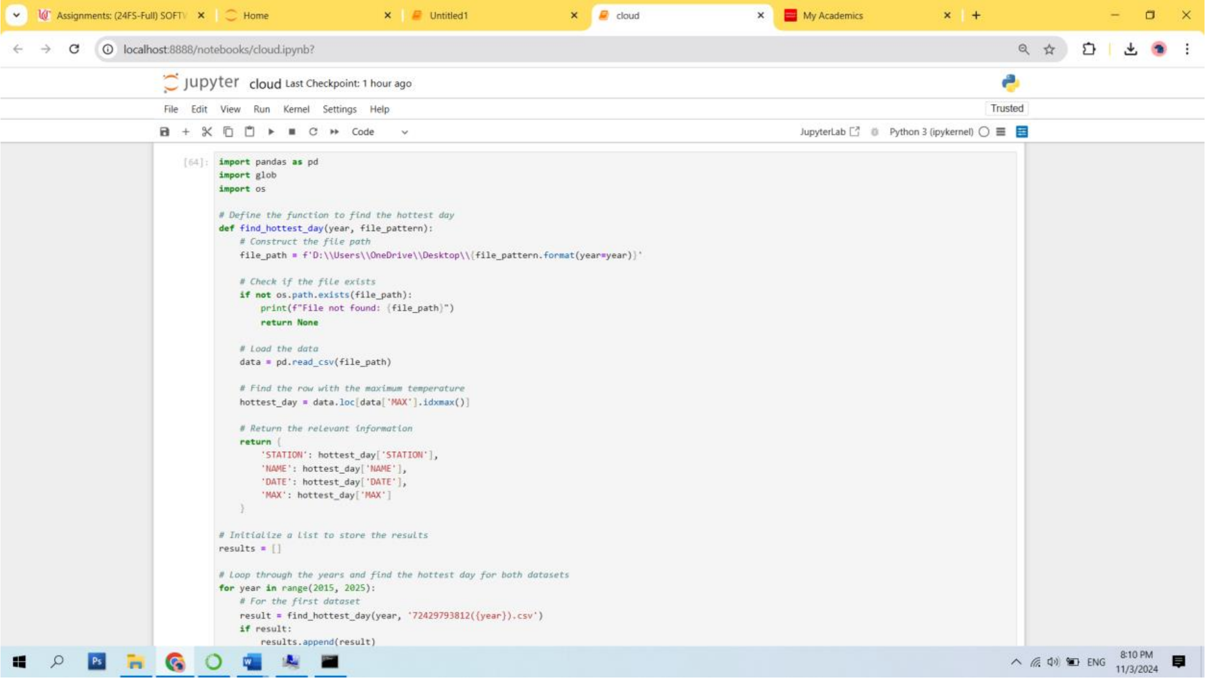
```
[58]: # Load each CSV and display counts
      cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2023).csv", header=True, inferSchema=True)
      florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2023).csv", header=True, inferSchema=True)

      # Display the counts
      print("Cincinnati Data Count:", cincinnati_data.count())
      print("Florida Data Count:", florida_data.count())
```

```
Cincinnati Data Count: 365
Florida Data Count: 276
```

```
[60]: # Load each CSV and display counts
      cincinnati_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\72429793812(2024).csv", header=True, inferSchema=True)
      florida_data = spark.read.csv("D:\\Users\\OneDrive\\Desktop\\99495199999(2024).csv", header=True, inferSchema=True)

      # Display the counts
      print("Cincinnati Data Count:", cincinnati_data.count())
      print("Florida Data Count:", florida_data.count())
```

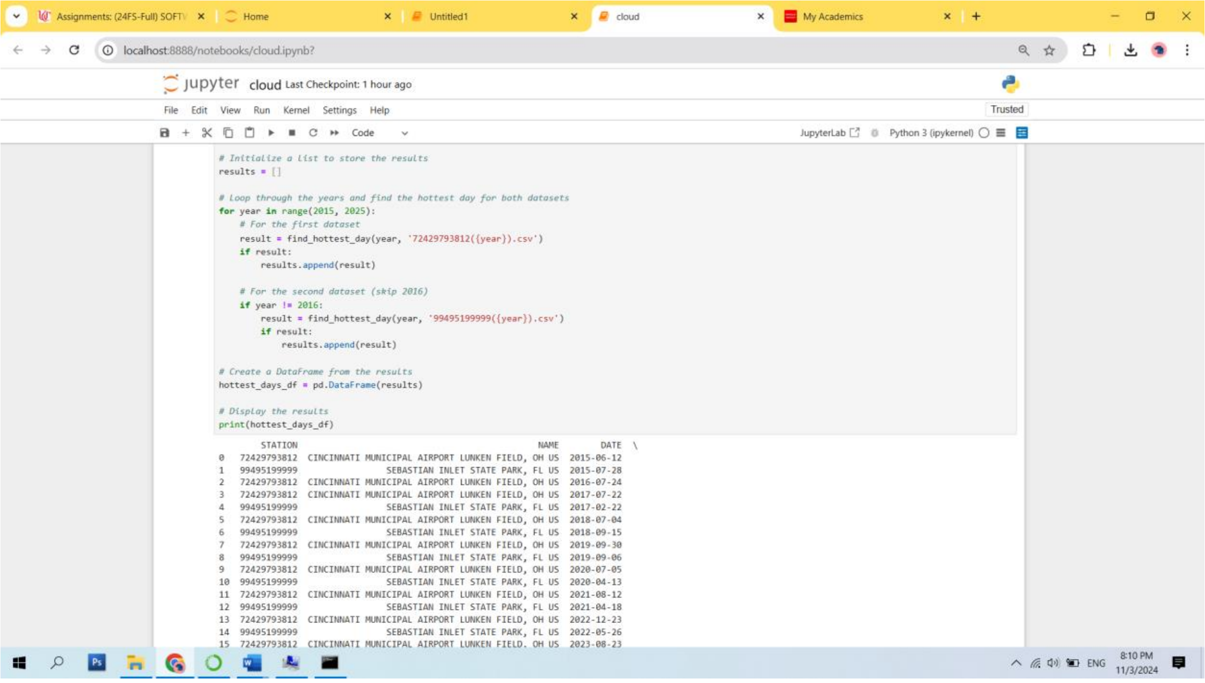
```
Cincinnati Data Count: 301
Florida Data Count: 133
```

3)



Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
8:10 PM
11/3/2024
ENG

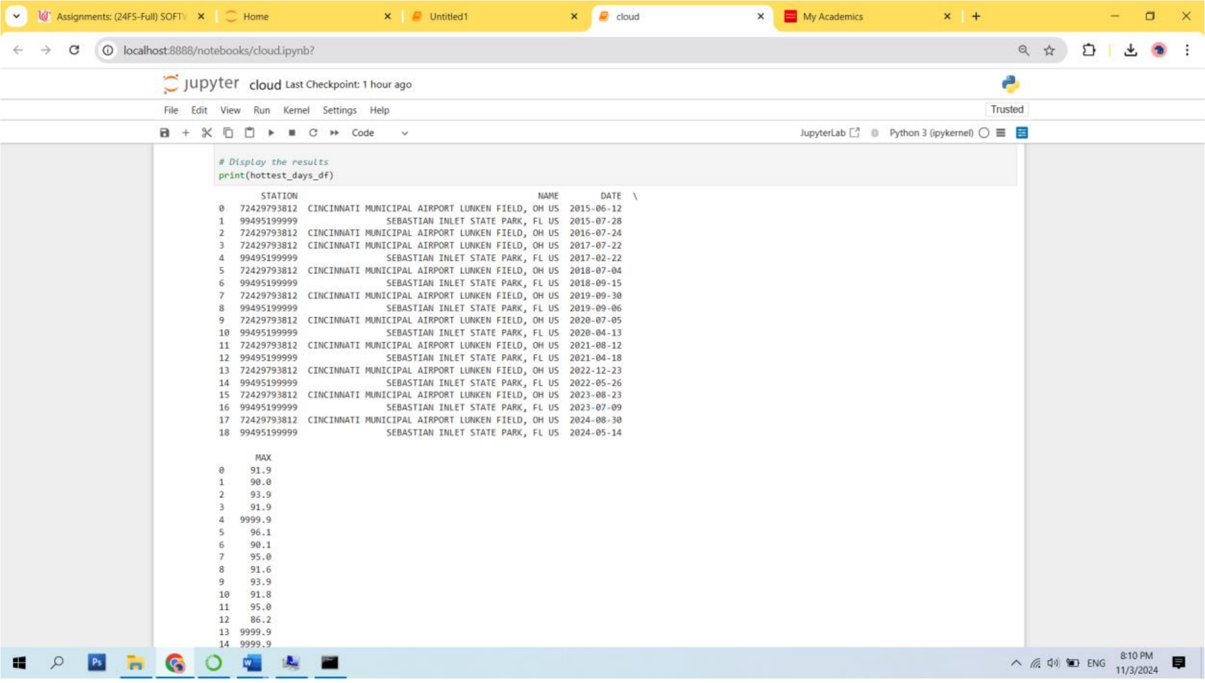
```
[64]: import pandas as pd
      import glob
      import os

      # Define the function to find the hottest day
      def find_hottest_day(year, file_pattern):
          # Construct the file path
          file_path = f'D:\\Users\\OneDrive\\Desktop\\{file_pattern.format(year=year)}'

          # Check if the file exists
          if not os.path.exists(file_path):
              print(f"File not found: {file_path}")
              return None

          # Load the data
          data = pd.read_csv(file_path)

          # Find the row with the maximum temperature
          hottest_day = data.loc[data['MAX'].idxmax()]

          # Return the relevant information
          return {
              'STATION': hottest_day['STATION'],
              'NAME': hottest_day['NAME'],
              'DATE': hottest_day['DATE'],
              'MAX': hottest_day['MAX']
          }

      # Initialize a list to store the results
      results = []

      # Loop through the years and find the hottest day for both datasets
      for year in range(2015, 2025):
          # For the first dataset
          result = find_hottest_day(year, '72429793812({year}).csv')
          if result:
              results.append(result)
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:10 PM
11/3/2024

```
# Initialize a list to store the results
results = []

# Loop through the years and find the hottest day for both datasets
for year in range(2015, 2025):
    # For the first dataset
    result = find_hottest_day(year, '72429793812({year}).csv')
    if result:
        results.append(result)

    # For the second dataset (skip 2016)
    if year != 2016:
        result = find_hottest_day(year, '99495199999({year}).csv')
        if result:
            results.append(result)

# Create a DataFrame from the results
hottest_days_df = pd.DataFrame(results)

# Display the results
print(hottest_days_df)
```

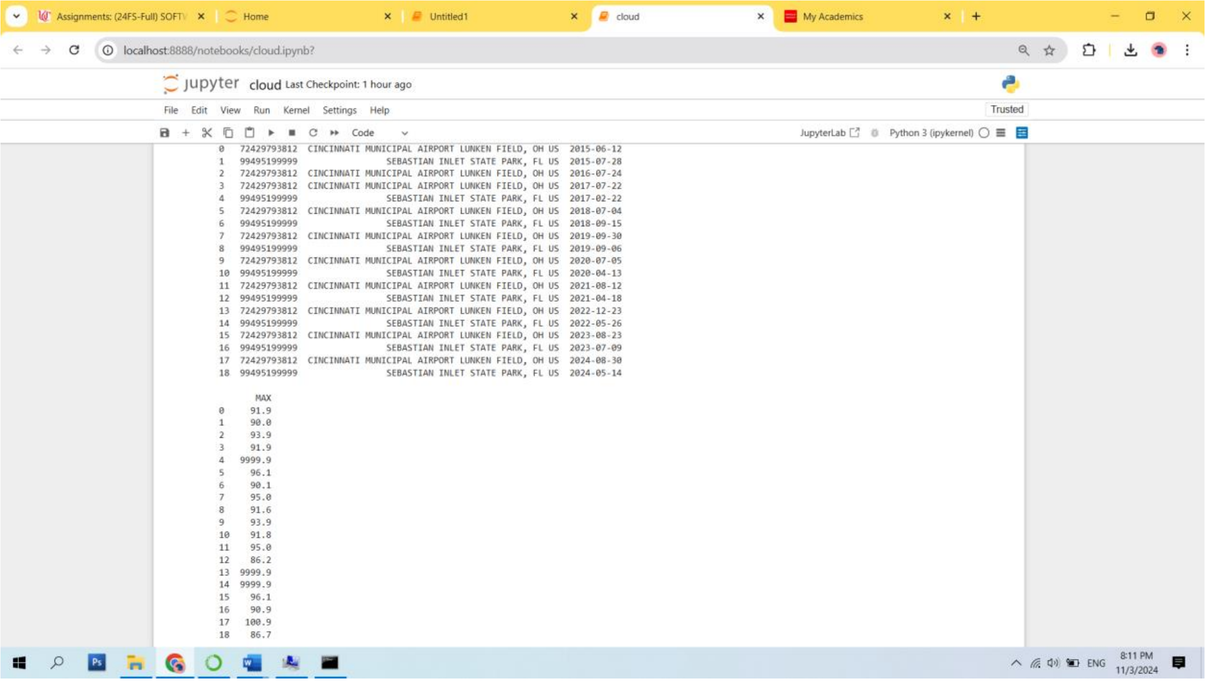
```
       STATION                                          NAME        DATE  \
0   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2015-06-12
1   99495199999                SEBASTIAN INLET STATE PARK, FL US  2015-07-28
2   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2016-07-24
3   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2017-07-22
4   99495199999                SEBASTIAN INLET STATE PARK, FL US  2017-02-22
5   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2018-07-04
6   99495199999                SEBASTIAN INLET STATE PARK, FL US  2018-09-15
7   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2019-09-30
8   99495199999                SEBASTIAN INLET STATE PARK, FL US  2019-09-06
9   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2020-07-05
10  99495199999                SEBASTIAN INLET STATE PARK, FL US  2020-04-13
11  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2021-08-12
12  99495199999                SEBASTIAN INLET STATE PARK, FL US  2021-04-18
13  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2022-12-23
14  99495199999                SEBASTIAN INLET STATE PARK, FL US  2022-05-26
15  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2023-08-23
```

4)


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:10 PM
11/3/2024

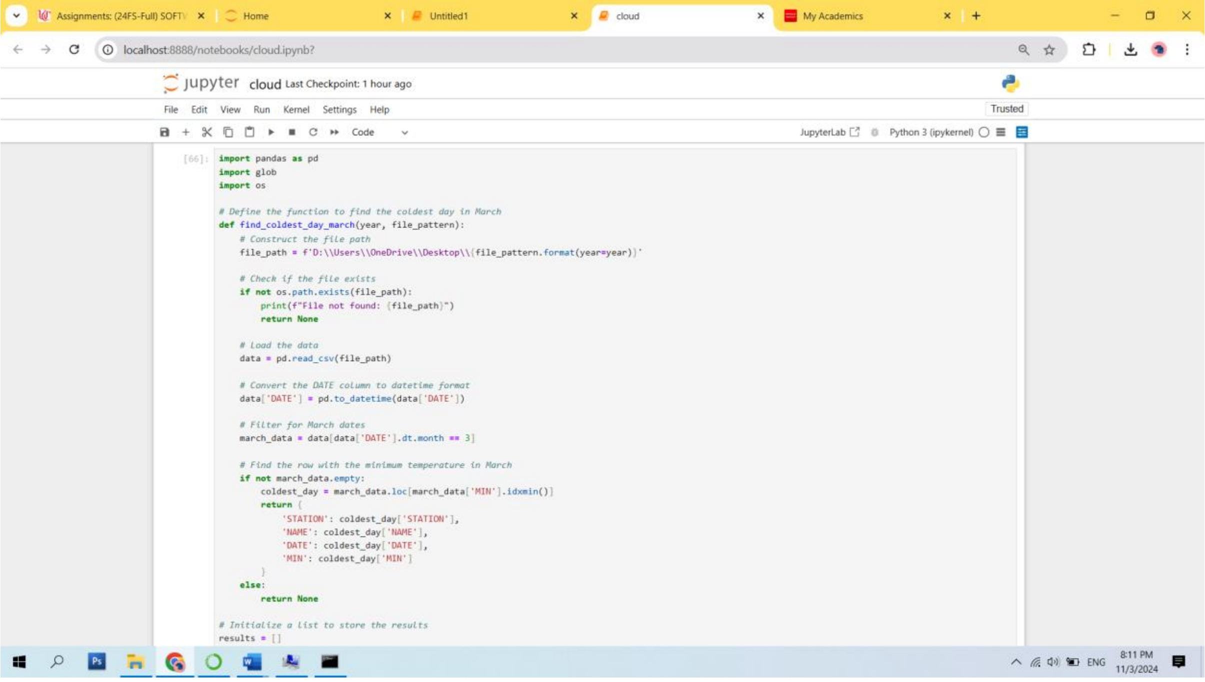
```
# Display the results
print(hottest_days_df)
```

```
       STATION                                          NAME        DATE  \
0   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2015-06-12
1   99495199999                SEBASTIAN INLET STATE PARK, FL US  2015-07-28
2   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2016-07-24
3   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2017-07-22
4   99495199999                SEBASTIAN INLET STATE PARK, FL US  2017-02-22
5   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2018-07-04
6   99495199999                SEBASTIAN INLET STATE PARK, FL US  2018-09-15
7   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2019-09-30
8   99495199999                SEBASTIAN INLET STATE PARK, FL US  2019-09-06
9   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2020-07-05
10  99495199999                SEBASTIAN INLET STATE PARK, FL US  2020-04-13
11  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2021-08-12
12  99495199999                SEBASTIAN INLET STATE PARK, FL US  2021-04-18
13  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2022-12-23
14  99495199999                SEBASTIAN INLET STATE PARK, FL US  2022-05-26
15  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2023-08-23
16  99495199999                SEBASTIAN INLET STATE PARK, FL US  2023-07-09
17  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2024-08-30
18  99495199999                SEBASTIAN INLET STATE PARK, FL US  2024-05-14

       MAX
0     91.9
1     90.0
2     93.9
3     91.9
4   9999.9
5     96.1
6     90.1
7     95.0
8     91.6
9     93.9
10    91.8
11    95.0
12    86.2
13  9999.9
14  9999.9
```

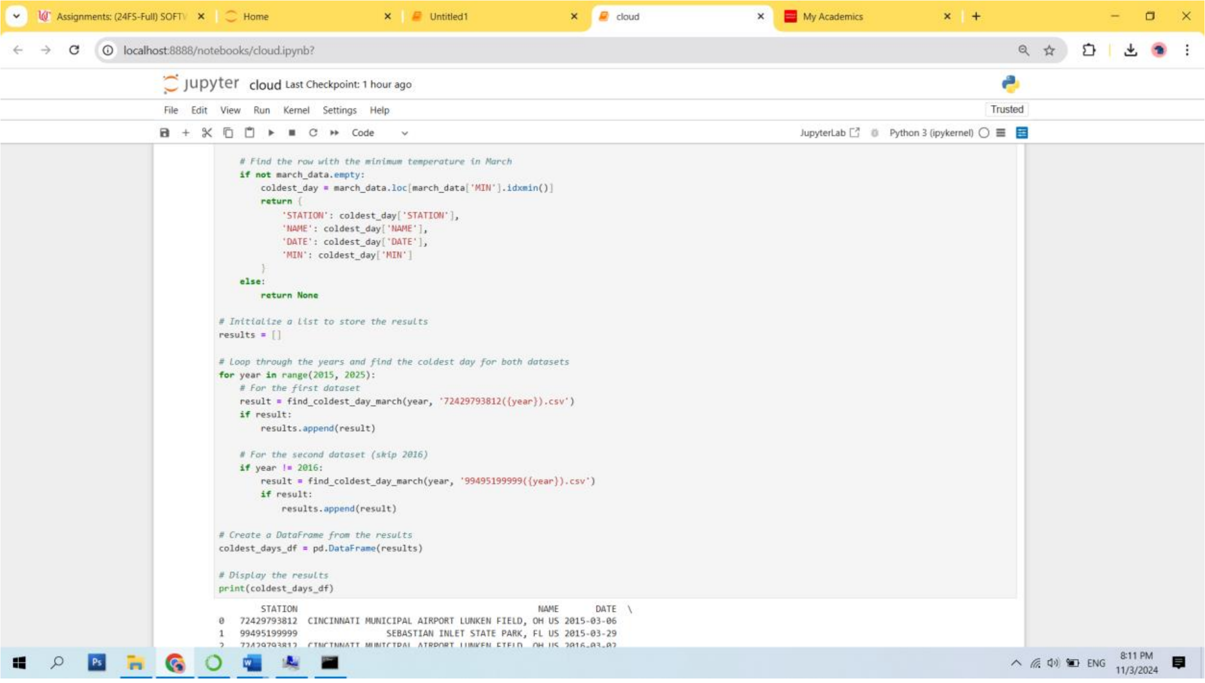

Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Code
Trusted
JupyterLab
Python 3 (ipykernel)
ENG
8:11 PM
11/3/2024

```
0   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2015-06-12
1   99495199999                    SEBASTIAN INLET STATE PARK, FL US  2015-07-28
2   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2016-07-24
3   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2017-07-22
4   99495199999                    SEBASTIAN INLET STATE PARK, FL US  2017-02-22
5   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2018-07-04
6   99495199999                    SEBASTIAN INLET STATE PARK, FL US  2018-09-15
7   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2019-09-30
8   99495199999                    SEBASTIAN INLET STATE PARK, FL US  2019-09-06
9   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2020-07-05
10  99495199999                    SEBASTIAN INLET STATE PARK, FL US  2020-04-13
11  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2021-08-12
12  99495199999                    SEBASTIAN INLET STATE PARK, FL US  2021-04-18
13  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2022-12-23
14  99495199999                    SEBASTIAN INLET STATE PARK, FL US  2022-05-26
15  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2023-08-23
16  99495199999                    SEBASTIAN INLET STATE PARK, FL US  2023-07-09
17  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US  2024-08-30
18  99495199999                    SEBASTIAN INLET STATE PARK, FL US  2024-05-14

       MAX
0     91.9
1     90.0
2     93.9
3     91.9
4   9999.9
5     96.1
6     90.1
7     95.0
8     91.6
9     93.9
10    91.8
11    95.0
12    86.2
13  9999.9
14  9999.9
15    96.1
16    90.9
17   100.9
18    86.7
```

5)

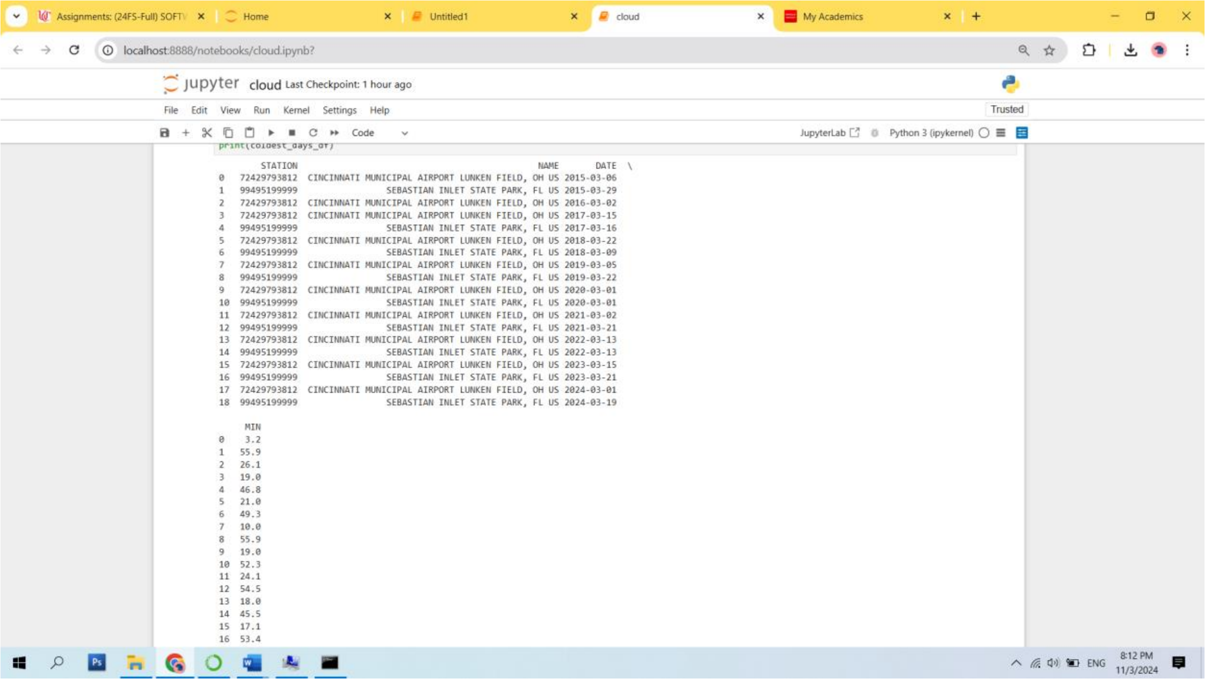

Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Code
Trusted
JupyterLab
Python 3 (ipykernel)
[66]:
ENG
8:11 PM
11/3/2024

```python
import pandas as pd
import glob
import os

# Define the function to find the coldest day in March
def find_coldest_day_march(year, file_pattern):
    # Construct the file path
    file_path = f'D:\\Users\\OneDrive\\Desktop\\{file_pattern.format(year=year)}'

    # Check if the file exists
    if not os.path.exists(file_path):
        print(f"File not found: {file_path}")
        return None

    # Load the data
    data = pd.read_csv(file_path)

    # Convert the DATE column to datetime format
    data['DATE'] = pd.to_datetime(data['DATE'])

    # Filter for March dates
    march_data = data[data['DATE'].dt.month == 3]

    # Find the row with the minimum temperature in March
    if not march_data.empty:
        coldest_day = march_data.loc[march_data['MIN'].idxmin()]
        return {
            'STATION': coldest_day['STATION'],
            'NAME': coldest_day['NAME'],
            'DATE': coldest_day['DATE'],
            'MIN': coldest_day['MIN']
        }
    else:
        return None

# Initialize a list to store the results
results = []
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:11 PM
11/3/2024

```
    # Find the row with the minimum temperature in March
    if not march_data.empty:
        coldest_day = march_data.loc[march_data['MIN'].idxmin()]
        return {
            'STATION': coldest_day['STATION'],
            'NAME': coldest_day['NAME'],
            'DATE': coldest_day['DATE'],
            'MIN': coldest_day['MIN']
        }
    else:
        return None

# Initialize a list to store the results
results = []

# Loop through the years and find the coldest day for both datasets
for year in range(2015, 2025):
    # For the first dataset
    result = find_coldest_day_march(year, '72429793812({year}).csv')
    if result:
        results.append(result)

    # For the second dataset (skip 2016)
    if year != 2016:
        result = find_coldest_day_march(year, '99495199999({year}).csv')
        if result:
            results.append(result)

# Create a DataFrame from the results
coldest_days_df = pd.DataFrame(results)

# Display the results
print(coldest_days_df)
```

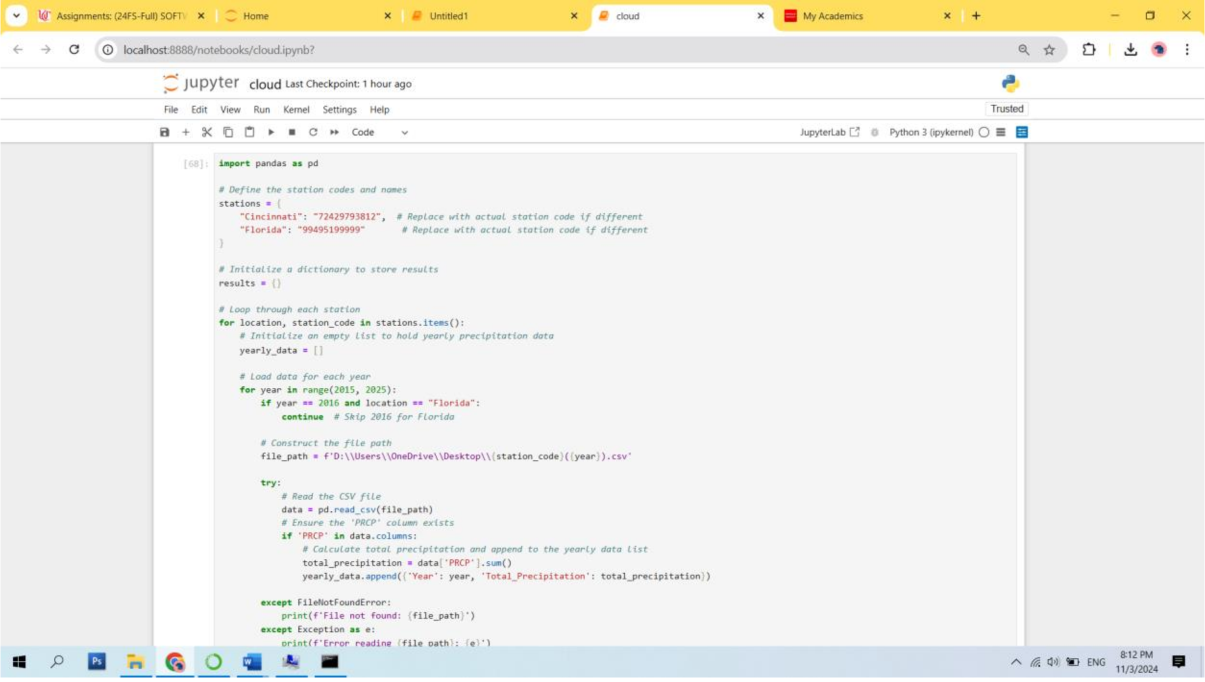
```
        STATION                                          NAME       DATE  \
0   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2015-03-06
1   99495199999                    SEBASTIAN INLET STATE PARK, FL US 2015-03-29
2   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2016-03-02
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:12 PM
11/3/2024

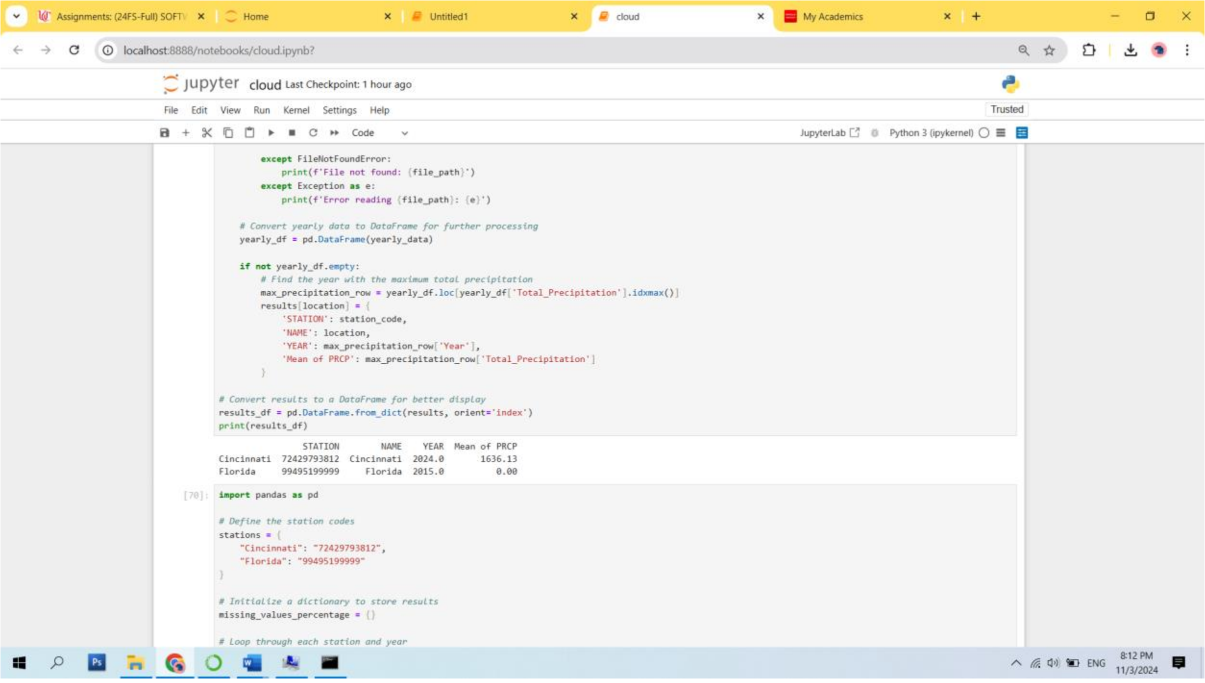
```
print(coldest_days_df)
```

```
        STATION                                          NAME       DATE  \
0   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2015-03-06
1   99495199999                    SEBASTIAN INLET STATE PARK, FL US 2015-03-29
2   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2016-03-02
3   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2017-03-15
4   99495199999                    SEBASTIAN INLET STATE PARK, FL US 2017-03-16
5   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2018-03-22
6   99495199999                    SEBASTIAN INLET STATE PARK, FL US 2018-03-09
7   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2019-03-05
8   99495199999                    SEBASTIAN INLET STATE PARK, FL US 2019-03-22
9   72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2020-03-01
10  99495199999                    SEBASTIAN INLET STATE PARK, FL US 2020-03-01
11  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2021-03-02
12  99495199999                    SEBASTIAN INLET STATE PARK, FL US 2021-03-21
13  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2022-03-13
14  99495199999                    SEBASTIAN INLET STATE PARK, FL US 2022-03-13
15  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2023-03-15
16  99495199999                    SEBASTIAN INLET STATE PARK, FL US 2023-03-21
17  72429793812  CINCINNATI MUNICIPAL AIRPORT LUNKEN FIELD, OH US 2024-03-01
18  99495199999                    SEBASTIAN INLET STATE PARK, FL US 2024-03-19

     MIN
0    3.2
1   55.9
2   26.1
3   19.0
4   46.8
5   21.0
6   49.3
7   10.0
8   55.9
9   19.0
10  52.3
11  24.1
12  54.5
13  18.0
14  45.5
15  17.1
16  53.4
```

6)


Assignments: (24FS-Full) SOFTW
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:12 PM
11/3/2024

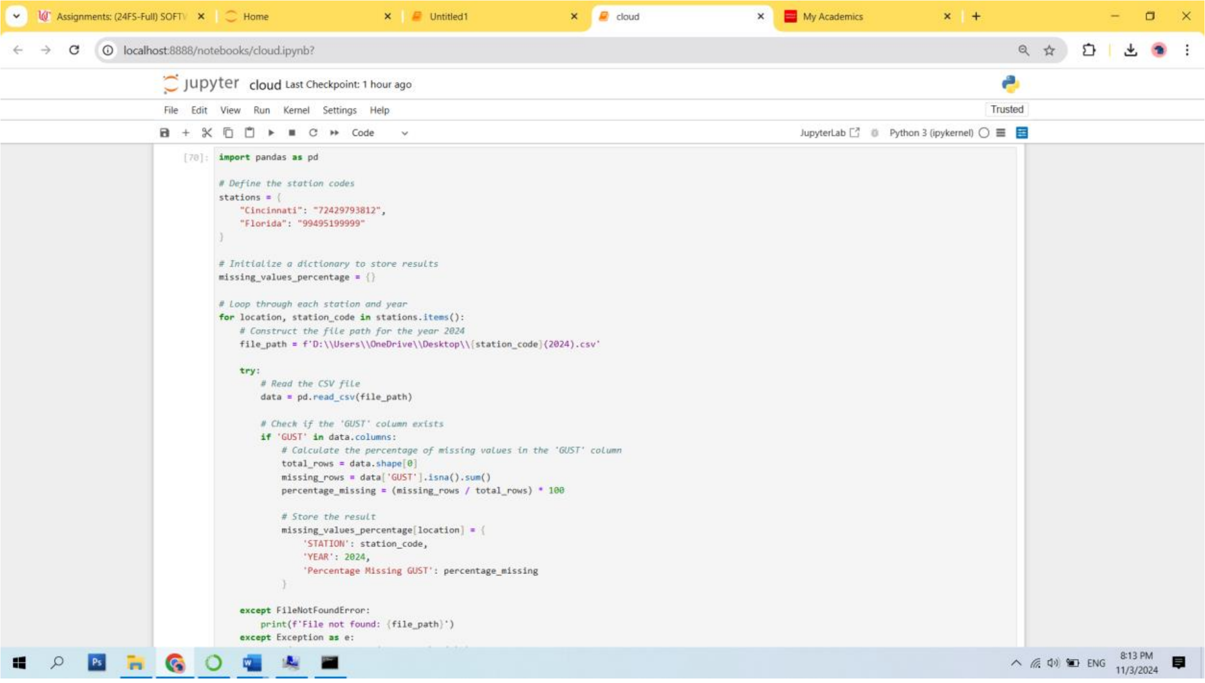
```
[68]: import pandas as pd

      # Define the station codes and names
      stations = {
          "Cincinnati": "72429793812",  # Replace with actual station code if different
          "Florida": "99495199999"       # Replace with actual station code if different
      }

      # Initialize a dictionary to store results
      results = {}

      # Loop through each station
      for location, station_code in stations.items():
          # Initialize an empty list to hold yearly precipitation data
          yearly_data = []

          # Load data for each year
          for year in range(2015, 2025):
              if year == 2016 and location == "Florida":
                  continue  # Skip 2016 for Florida

              # Construct the file path
              file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}({year}).csv'

              try:
                  # Read the CSV file
                  data = pd.read_csv(file_path)
                  # Ensure the 'PRCP' column exists
                  if 'PRCP' in data.columns:
                      # Calculate total precipitation and append to the yearly data list
                      total_precipitation = data['PRCP'].sum()
                      yearly_data.append({'Year': year, 'Total_Precipitation': total_precipitation})

              except FileNotFoundError:
                  print(f'File not found: {file_path}')
              except Exception as e:
                  print(f'Error reading {file_path}: {e}')
```


Assignments: (24FS-Full) SOFTW
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:12 PM
11/3/2024

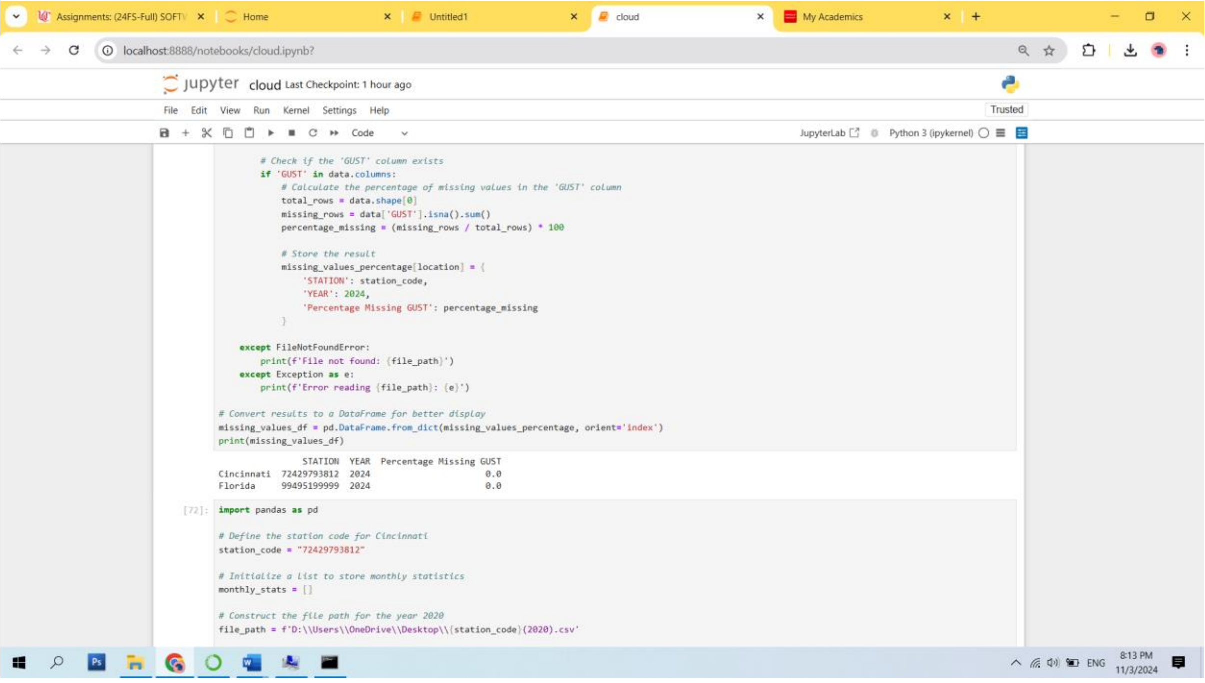
```
              except FileNotFoundError:
                  print(f'File not found: {file_path}')
              except Exception as e:
                  print(f'Error reading {file_path}: {e}')

          # Convert yearly data to DataFrame for further processing
          yearly_df = pd.DataFrame(yearly_data)

          if not yearly_df.empty:
              # Find the year with the maximum total precipitation
              max_precipitation_row = yearly_df.loc[yearly_df['Total_Precipitation'].idxmax()]
              results[location] = {
                  'STATION': station_code,
                  'NAME': location,
                  'YEAR': max_precipitation_row['Year'],
                  'Mean of PRCP': max_precipitation_row['Total_Precipitation']
              }

      # Convert results to a DataFrame for better display
      results_df = pd.DataFrame.from_dict(results, orient='index')
      print(results_df)
```

```
                STATION        NAME    YEAR  Mean of PRCP
Cincinnati  72429793812  Cincinnati  2024.0       1636.13
Florida     99495199999     Florida  2015.0          0.00
```

```
[70]: import pandas as pd

      # Define the station codes
      stations = {
          "Cincinnati": "72429793812",
          "Florida": "99495199999"
      }

      # Initialize a dictionary to store results
      missing_values_percentage = {}

      # Loop through each station and year
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:13 PM
11/3/2024

```
[70]: import pandas as pd

      # Define the station codes
      stations = {
          "Cincinnati": "72429793812",
          "Florida": "99495199999"
      }

      # Initialize a dictionary to store results
      missing_values_percentage = {}

      # Loop through each station and year
      for location, station_code in stations.items():
          # Construct the file path for the year 2024
          file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}(2024).csv'

          try:
              # Read the CSV file
              data = pd.read_csv(file_path)

              # Check if the 'GUST' column exists
              if 'GUST' in data.columns:
                  # Calculate the percentage of missing values in the 'GUST' column
                  total_rows = data.shape[0]
                  missing_rows = data['GUST'].isna().sum()
                  percentage_missing = (missing_rows / total_rows) * 100

                  # Store the result
                  missing_values_percentage[location] = {
                      'STATION': station_code,
                      'YEAR': 2024,
                      'Percentage Missing GUST': percentage_missing
                  }

          except FileNotFoundError:
              print(f'File not found: {file_path}')
          except Exception as e:
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:13 PM
11/3/2024

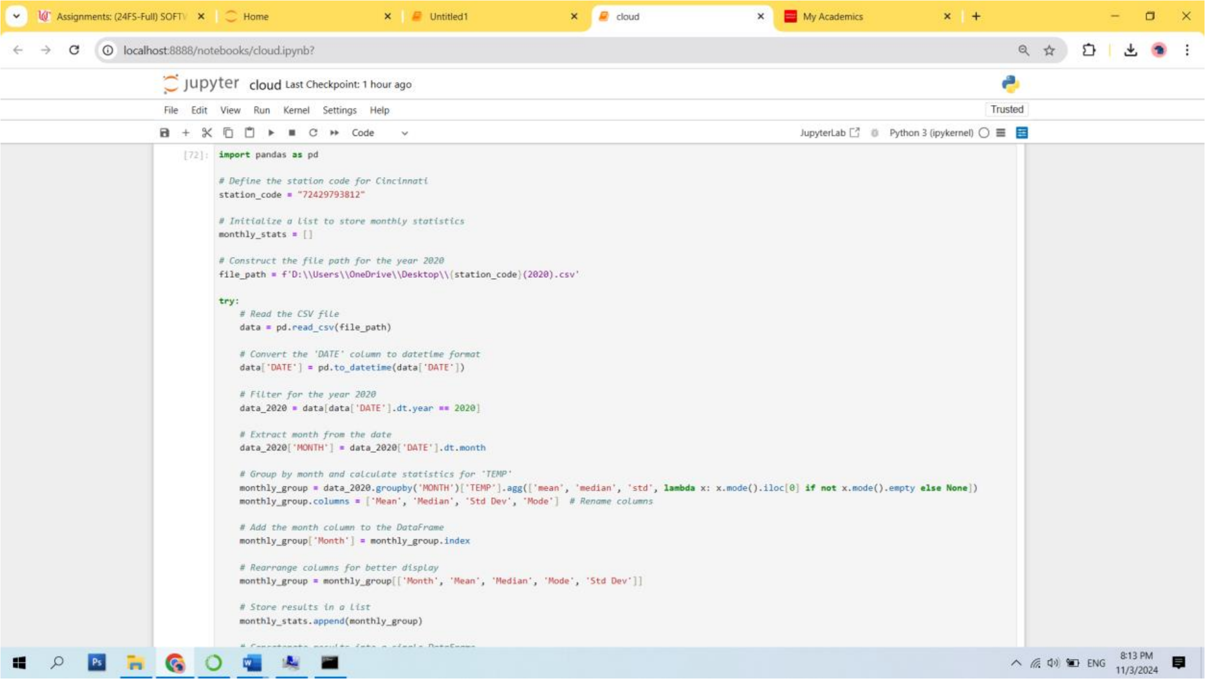
```
              # Check if the 'GUST' column exists
              if 'GUST' in data.columns:
                  # Calculate the percentage of missing values in the 'GUST' column
                  total_rows = data.shape[0]
                  missing_rows = data['GUST'].isna().sum()
                  percentage_missing = (missing_rows / total_rows) * 100

                  # Store the result
                  missing_values_percentage[location] = {
                      'STATION': station_code,
                      'YEAR': 2024,
                      'Percentage Missing GUST': percentage_missing
                  }

          except FileNotFoundError:
              print(f'File not found: {file_path}')
          except Exception as e:
```

```
              print(f'Error reading {file_path}: {e}')

      # Convert results to a DataFrame for better display
      missing_values_df = pd.DataFrame.from_dict(missing_values_percentage, orient='index')
      print(missing_values_df)
```

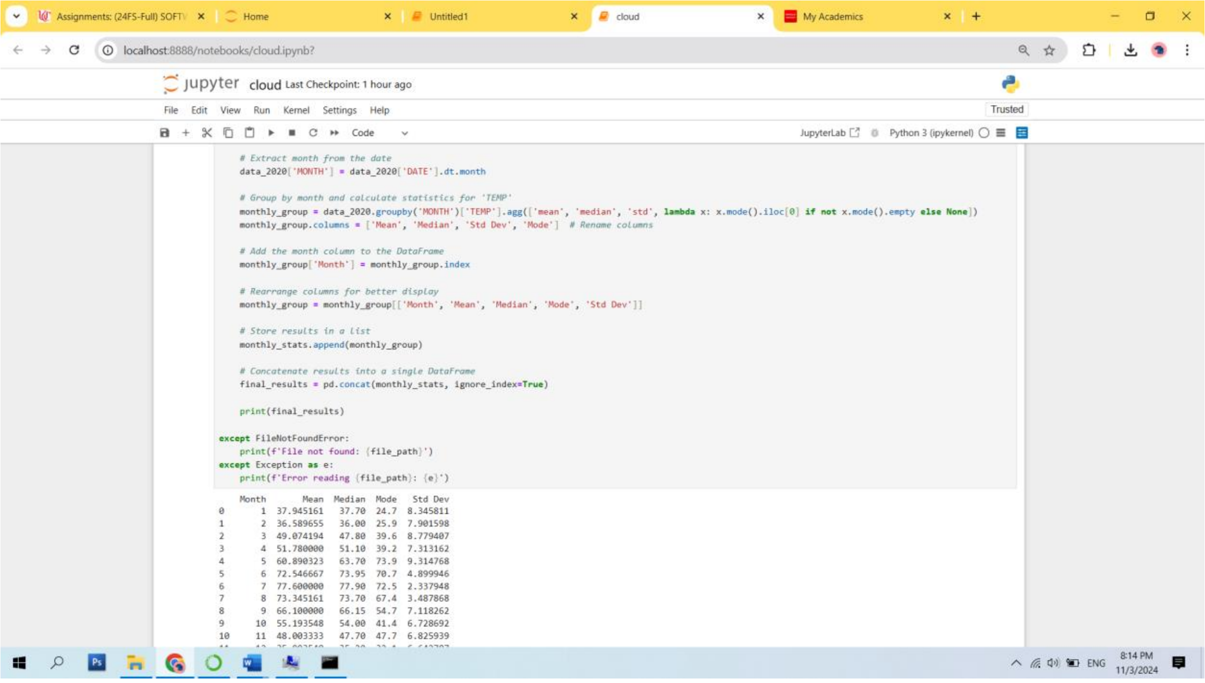
```
                STATION  YEAR  Percentage Missing GUST
Cincinnati  72429793812  2024                      0.0
Florida     99495199999  2024                      0.0
```

```
[72]: import pandas as pd

      # Define the station code for Cincinnati
      station_code = "72429793812"

      # Initialize a list to store monthly statistics
      monthly_stats = []

      # Construct the file path for the year 2020
      file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}(2020).csv'
```

7)

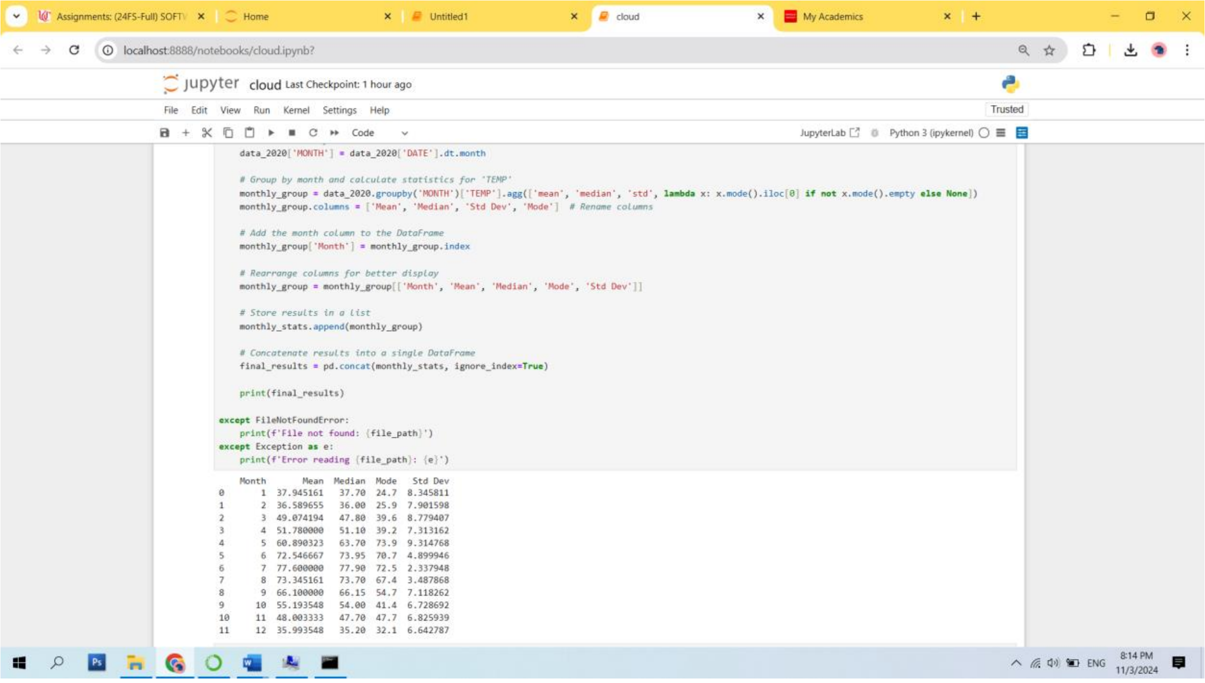

Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:13 PM
11/3/2024

```
[72]: import pandas as pd

      # Define the station code for Cincinnati
      station_code = "72429793812"

      # Initialize a list to store monthly statistics
      monthly_stats = []

      # Construct the file path for the year 2020
      file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}(2020).csv'

      try:
          # Read the CSV file
          data = pd.read_csv(file_path)

          # Convert the 'DATE' column to datetime format
          data['DATE'] = pd.to_datetime(data['DATE'])

          # Filter for the year 2020
          data_2020 = data[data['DATE'].dt.year == 2020]

          # Extract month from the date
          data_2020['MONTH'] = data_2020['DATE'].dt.month

          # Group by month and calculate statistics for 'TEMP'
          monthly_group = data_2020.groupby('MONTH')['TEMP'].agg(['mean', 'median', 'std', lambda x: x.mode().iloc[0] if not x.mode().empty else None])
          monthly_group.columns = ['Mean', 'Median', 'Std Dev', 'Mode']  # Rename columns

          # Add the month column to the DataFrame
          monthly_group['Month'] = monthly_group.index

          # Rearrange columns for better display
          monthly_group = monthly_group[['Month', 'Mean', 'Median', 'Mode', 'Std Dev']]

          # Store results in a list
          monthly_stats.append(monthly_group)
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:14 PM
11/3/2024

```
    # Extract month from the date
    data_2020['MONTH'] = data_2020['DATE'].dt.month

    # Group by month and calculate statistics for 'TEMP'
    monthly_group = data_2020.groupby('MONTH')['TEMP'].agg(['mean', 'median', 'std', lambda x: x.mode().iloc[0] if not x.mode().empty else None])
    monthly_group.columns = ['Mean', 'Median', 'Std Dev', 'Mode']  # Rename columns

    # Add the month column to the DataFrame
    monthly_group['Month'] = monthly_group.index

    # Rearrange columns for better display
    monthly_group = monthly_group[['Month', 'Mean', 'Median', 'Mode', 'Std Dev']]

    # Store results in a list
    monthly_stats.append(monthly_group)

    # Concatenate results into a single DataFrame
    final_results = pd.concat(monthly_stats, ignore_index=True)

    print(final_results)

except FileNotFoundError:
    print(f'File not found: {file_path}')
except Exception as e:
    print(f'Error reading {file_path}: {e}')
```

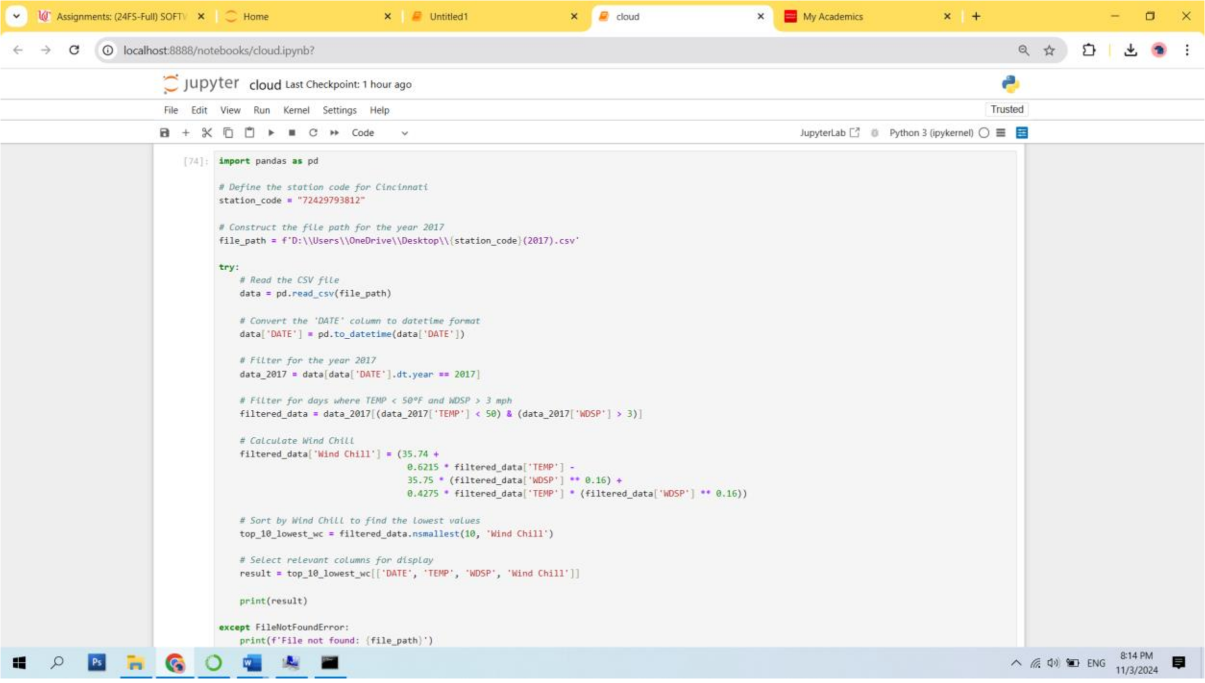
```
    Month       Mean  Median  Mode   Std Dev
0       1  37.945161   37.70  24.7  8.345811
1       2  36.589655   36.00  25.9  7.901598
2       3  49.074194   47.80  39.6  8.779407
3       4  51.780000   51.10  39.2  7.313162
4       5  60.890323   63.70  73.9  9.314768
5       6  72.546667   73.95  70.7  4.899946
6       7  77.600000   77.90  72.5  2.337948
7       8  73.345161   73.70  67.4  3.487868
8       9  66.100000   66.15  54.7  7.118262
9      10  55.193548   54.00  41.4  6.728692
10     11  48.003333   47.70  47.7  6.825939
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:14 PM
11/3/2024

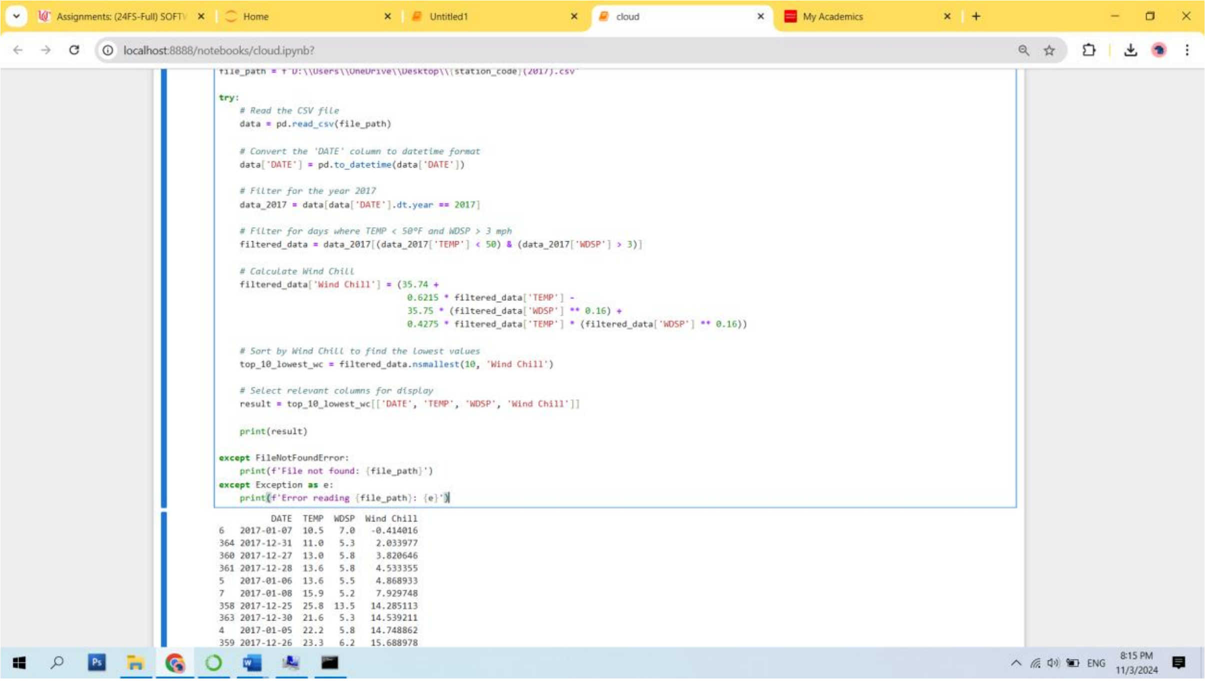
```
    data_2020['MONTH'] = data_2020['DATE'].dt.month

    # Group by month and calculate statistics for 'TEMP'
    monthly_group = data_2020.groupby('MONTH')['TEMP'].agg(['mean', 'median', 'std', lambda x: x.mode().iloc[0] if not x.mode().empty else None])
    monthly_group.columns = ['Mean', 'Median', 'Std Dev', 'Mode']  # Rename columns

    # Add the month column to the DataFrame
    monthly_group['Month'] = monthly_group.index

    # Rearrange columns for better display
    monthly_group = monthly_group[['Month', 'Mean', 'Median', 'Mode', 'Std Dev']]

    # Store results in a list
    monthly_stats.append(monthly_group)

    # Concatenate results into a single DataFrame
    final_results = pd.concat(monthly_stats, ignore_index=True)

    print(final_results)

except FileNotFoundError:
    print(f'File not found: {file_path}')
except Exception as e:
    print(f'Error reading {file_path}: {e}')
```

```
    Month       Mean  Median  Mode   Std Dev
0       1  37.945161   37.70  24.7  8.345811
1       2  36.589655   36.00  25.9  7.901598
2       3  49.074194   47.80  39.6  8.779407
3       4  51.780000   51.10  39.2  7.313162
4       5  60.890323   63.70  73.9  9.314768
5       6  72.546667   73.95  70.7  4.899946
6       7  77.600000   77.90  72.5  2.337948
7       8  73.345161   73.70  67.4  3.487868
8       9  66.100000   66.15  54.7  7.118262
9      10  55.193548   54.00  41.4  6.728692
10     11  48.003333   47.70  47.7  6.825939
11     12  35.993548   35.20  32.1  6.642787
```

8)


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
jupyter cloud Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ENG
8:14 PM
11/3/2024

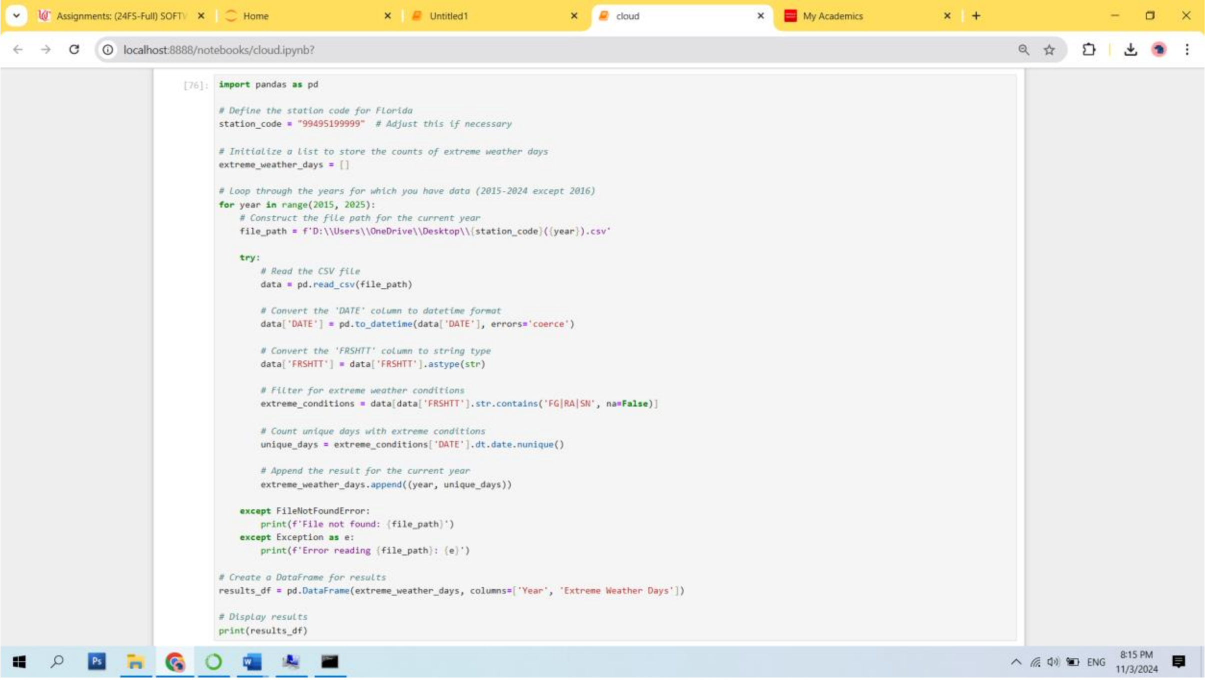
```
[74]: import pandas as pd

# Define the station code for Cincinnati
station_code = "72429793812"

# Construct the file path for the year 2017
file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}(2017).csv'

try:
    # Read the CSV file
    data = pd.read_csv(file_path)

    # Convert the 'DATE' column to datetime format
    data['DATE'] = pd.to_datetime(data['DATE'])

    # Filter for the year 2017
    data_2017 = data[data['DATE'].dt.year == 2017]

    # Filter for days where TEMP < 50°F and WDSP > 3 mph
    filtered_data = data_2017[(data_2017['TEMP'] < 50) & (data_2017['WDSP'] > 3)]

    # Calculate Wind Chill
    filtered_data['Wind Chill'] = (35.74 +
                                   0.6215 * filtered_data['TEMP'] -
                                   35.75 * (filtered_data['WDSP'] ** 0.16) +
                                   0.4275 * filtered_data['TEMP'] * (filtered_data['WDSP'] ** 0.16))

    # Sort by Wind Chill to find the lowest values
    top_10_lowest_wc = filtered_data.nsmallest(10, 'Wind Chill')

    # Select relevant columns for display
    result = top_10_lowest_wc[['DATE', 'TEMP', 'WDSP', 'Wind Chill']]

    print(result)

except FileNotFoundError:
    print(f'File not found: {file_path}')
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
ENG
8:15 PM
11/3/2024

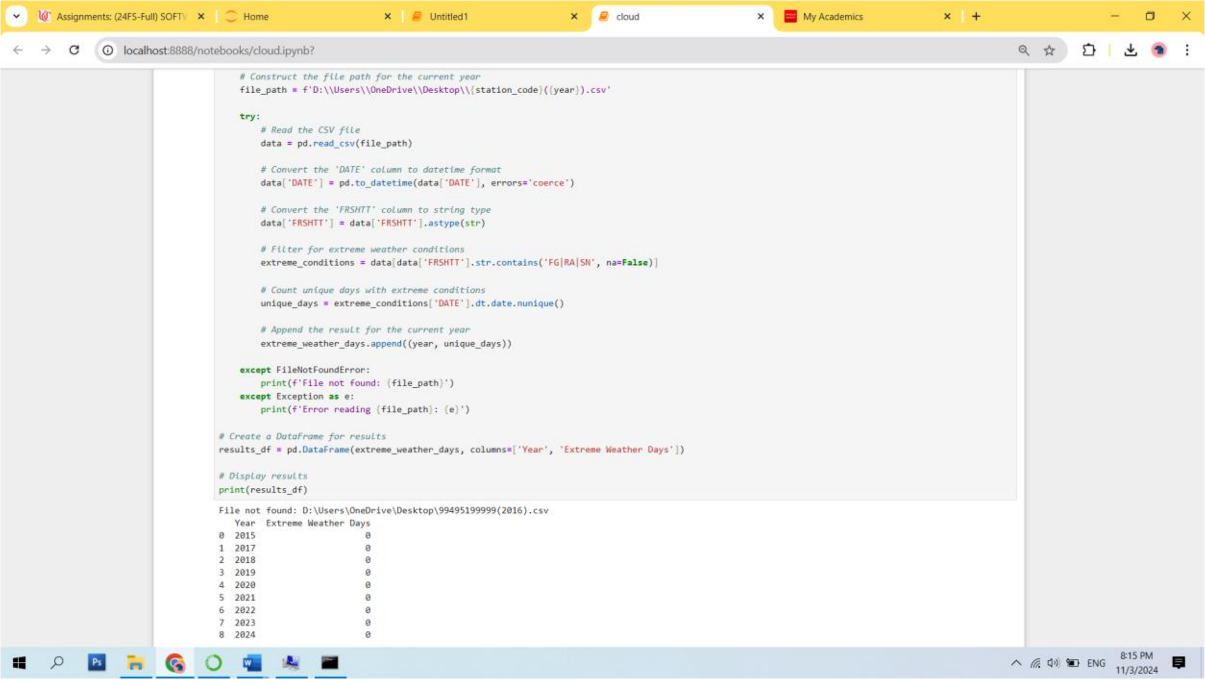
```
file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}(2017).csv'

try:
    # Read the CSV file
    data = pd.read_csv(file_path)

    # Convert the 'DATE' column to datetime format
    data['DATE'] = pd.to_datetime(data['DATE'])

    # Filter for the year 2017
    data_2017 = data[data['DATE'].dt.year == 2017]

    # Filter for days where TEMP < 50°F and WDSP > 3 mph
    filtered_data = data_2017[(data_2017['TEMP'] < 50) & (data_2017['WDSP'] > 3)]

    # Calculate Wind Chill
    filtered_data['Wind Chill'] = (35.74 +
                                   0.6215 * filtered_data['TEMP'] -
                                   35.75 * (filtered_data['WDSP'] ** 0.16) +
                                   0.4275 * filtered_data['TEMP'] * (filtered_data['WDSP'] ** 0.16))

    # Sort by Wind Chill to find the lowest values
    top_10_lowest_wc = filtered_data.nsmallest(10, 'Wind Chill')

    # Select relevant columns for display
    result = top_10_lowest_wc[['DATE', 'TEMP', 'WDSP', 'Wind Chill']]

    print(result)

except FileNotFoundError:
    print(f'File not found: {file_path}')
except Exception as e:
    print(f'Error reading {file_path}: {e}')
```

```
          DATE  TEMP  WDSP  Wind Chill
6   2017-01-07  10.5   7.0   -0.414016
364 2017-12-31  11.0   5.3    2.033977
360 2017-12-27  13.0   5.8    3.820646
361 2017-12-28  13.6   5.8    4.533355
5   2017-01-06  13.6   5.5    4.868933
7   2017-01-08  15.9   5.2    7.929748
358 2017-12-25  25.8  13.5   14.285113
363 2017-12-30  21.6   5.3   14.539211
4   2017-01-05  22.2   5.8   14.748862
359 2017-12-26  23.3   6.2   15.688978
```

9)


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
[76]:
ENG
8:15 PM
11/3/2024

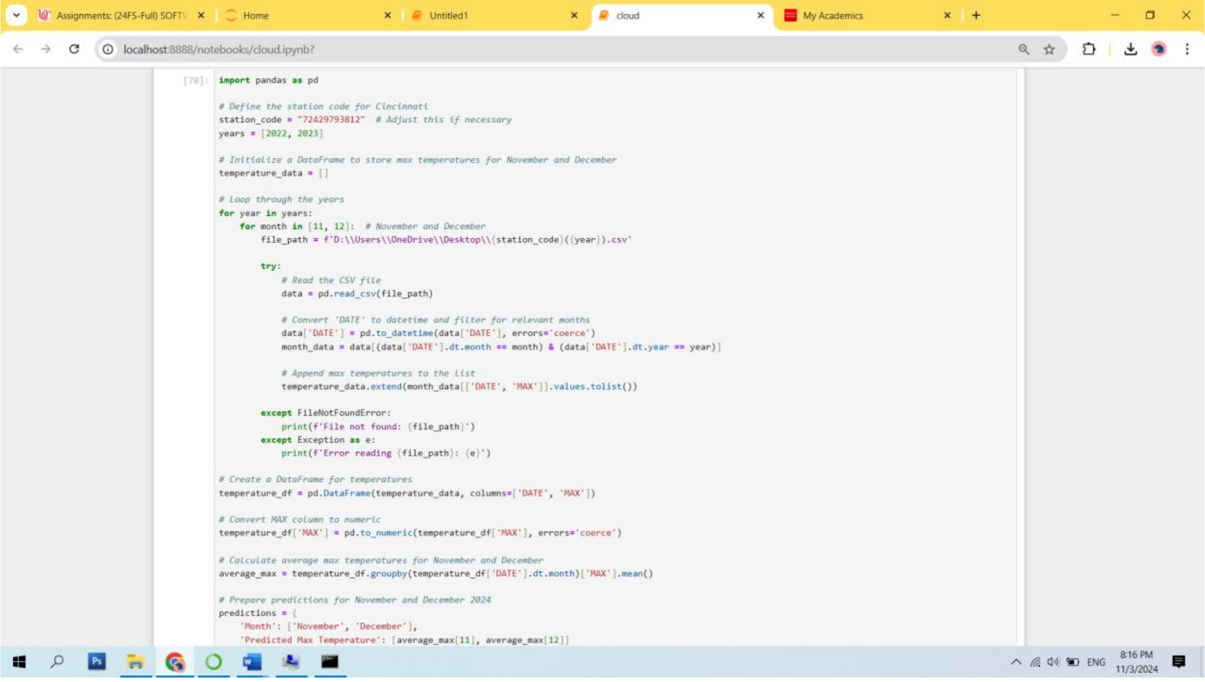
```
import pandas as pd

# Define the station code for Florida
station_code = "99495199999"  # Adjust this if necessary

# Initialize a list to store the counts of extreme weather days
extreme_weather_days = []

# Loop through the years for which you have data (2015-2024 except 2016)
for year in range(2015, 2025):
    # Construct the file path for the current year
    file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}({year}).csv'

    try:
        # Read the CSV file
        data = pd.read_csv(file_path)

        # Convert the 'DATE' column to datetime format
        data['DATE'] = pd.to_datetime(data['DATE'], errors='coerce')

        # Convert the 'FRSHTT' column to string type
        data['FRSHTT'] = data['FRSHTT'].astype(str)

        # Filter for extreme weather conditions
        extreme_conditions = data[data['FRSHTT'].str.contains('FG|RA|SN', na=False)]

        # Count unique days with extreme conditions
        unique_days = extreme_conditions['DATE'].dt.date.nunique()

        # Append the result for the current year
        extreme_weather_days.append((year, unique_days))

    except FileNotFoundError:
        print(f'File not found: {file_path}')
    except Exception as e:
        print(f'Error reading {file_path}: {e}')

# Create a DataFrame for results
results_df = pd.DataFrame(extreme_weather_days, columns=['Year', 'Extreme Weather Days'])

# Display results
print(results_df)
```


Assignments: (24FS-Full) SOFT
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
ENG
8:15 PM
11/3/2024

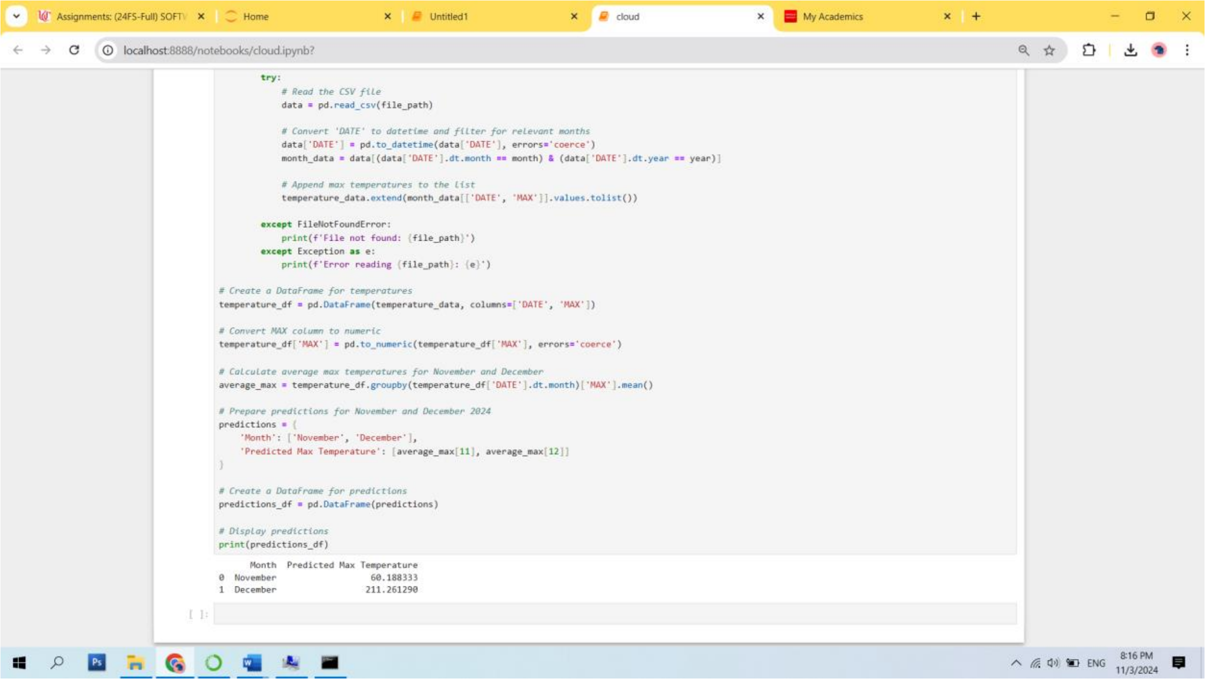
```
    # Construct the file path for the current year
    file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}({year}).csv'

    try:
        # Read the CSV file
        data = pd.read_csv(file_path)

        # Convert the 'DATE' column to datetime format
        data['DATE'] = pd.to_datetime(data['DATE'], errors='coerce')

        # Convert the 'FRSHTT' column to string type
        data['FRSHTT'] = data['FRSHTT'].astype(str)

        # Filter for extreme weather conditions
        extreme_conditions = data[data['FRSHTT'].str.contains('FG|RA|SN', na=False)]

        # Count unique days with extreme conditions
        unique_days = extreme_conditions['DATE'].dt.date.nunique()

        # Append the result for the current year
        extreme_weather_days.append((year, unique_days))

    except FileNotFoundError:
        print(f'File not found: {file_path}')
    except Exception as e:
        print(f'Error reading {file_path}: {e}')

# Create a DataFrame for results
results_df = pd.DataFrame(extreme_weather_days, columns=['Year', 'Extreme Weather Days'])

# Display results
print(results_df)
```

```
File not found: D:\Users\OneDrive\Desktop\99495199999(2016).csv
   Year  Extreme Weather Days
0  2015                     0
1  2017                     0
2  2018                     0
3  2019                     0
4  2020                     0
5  2021                     0
6  2022                     0
7  2023                     0
8  2024                     0
```

10)


Assignments: (24FS-Full) SOFT
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
ENG
8:16 PM
11/3/2024

```
[78]: import pandas as pd

      # Define the station code for Cincinnati
      station_code = "72429793812"  # Adjust this if necessary
      years = [2022, 2023]

      # Initialize a DataFrame to store max temperatures for November and December
      temperature_data = []

      # Loop through the years
      for year in years:
          for month in [11, 12]:  # November and December
              file_path = f'D:\\Users\\OneDrive\\Desktop\\{station_code}({year}).csv'

              try:
                  # Read the CSV file
                  data = pd.read_csv(file_path)

                  # Convert 'DATE' to datetime and filter for relevant months
                  data['DATE'] = pd.to_datetime(data['DATE'], errors='coerce')
                  month_data = data[(data['DATE'].dt.month == month) & (data['DATE'].dt.year == year)]

                  # Append max temperatures to the list
                  temperature_data.extend(month_data[['DATE', 'MAX']].values.tolist())

              except FileNotFoundError:
                  print(f'File not found: {file_path}')
              except Exception as e:
                  print(f'Error reading {file_path}: {e}')

      # Create a DataFrame for temperatures
      temperature_df = pd.DataFrame(temperature_data, columns=['DATE', 'MAX'])

      # Convert MAX column to numeric
      temperature_df['MAX'] = pd.to_numeric(temperature_df['MAX'], errors='coerce')

      # Calculate average max temperatures for November and December
      average_max = temperature_df.groupby(temperature_df['DATE'].dt.month)['MAX'].mean()

      # Prepare predictions for November and December 2024
      predictions = {
          'Month': ['November', 'December'],
          'Predicted Max Temperature': [average_max[11], average_max[12]]
```


Assignments: (24FS-Full) SOFTV
Home
Untitled1
cloud
My Academics
localhost:8888/notebooks/cloud.ipynb?
ENG
8:16 PM
11/3/2024

```
        try:
            # Read the CSV file
            data = pd.read_csv(file_path)

            # Convert 'DATE' to datetime and filter for relevant months
            data['DATE'] = pd.to_datetime(data['DATE'], errors='coerce')
            month_data = data[(data['DATE'].dt.month == month) & (data['DATE'].dt.year == year)]

            # Append max temperatures to the list
            temperature_data.extend(month_data[['DATE', 'MAX']].values.tolist())

        except FileNotFoundError:
            print(f'File not found: {file_path}')
        except Exception as e:
            print(f'Error reading {file_path}: {e}')

# Create a DataFrame for temperatures
temperature_df = pd.DataFrame(temperature_data, columns=['DATE', 'MAX'])

# Convert MAX column to numeric
temperature_df['MAX'] = pd.to_numeric(temperature_df['MAX'], errors='coerce')

# Calculate average max temperatures for November and December
average_max = temperature_df.groupby(temperature_df['DATE'].dt.month)['MAX'].mean()

# Prepare predictions for November and December 2024
predictions = {
    'Month': ['November', 'December'],
    'Predicted Max Temperature': [average_max[11], average_max[12]]
}

# Create a DataFrame for predictions
predictions_df = pd.DataFrame(predictions)

# Display predictions
print(predictions_df)
```

```
      Month  Predicted Max Temperature
0  November                  60.188333
1  December                 211.261290
```

```
[ ]:
```

M16561857 – Niharika Mysore Gowda

Niharika Mysore Gowda

College Of Engineering and Applied Science