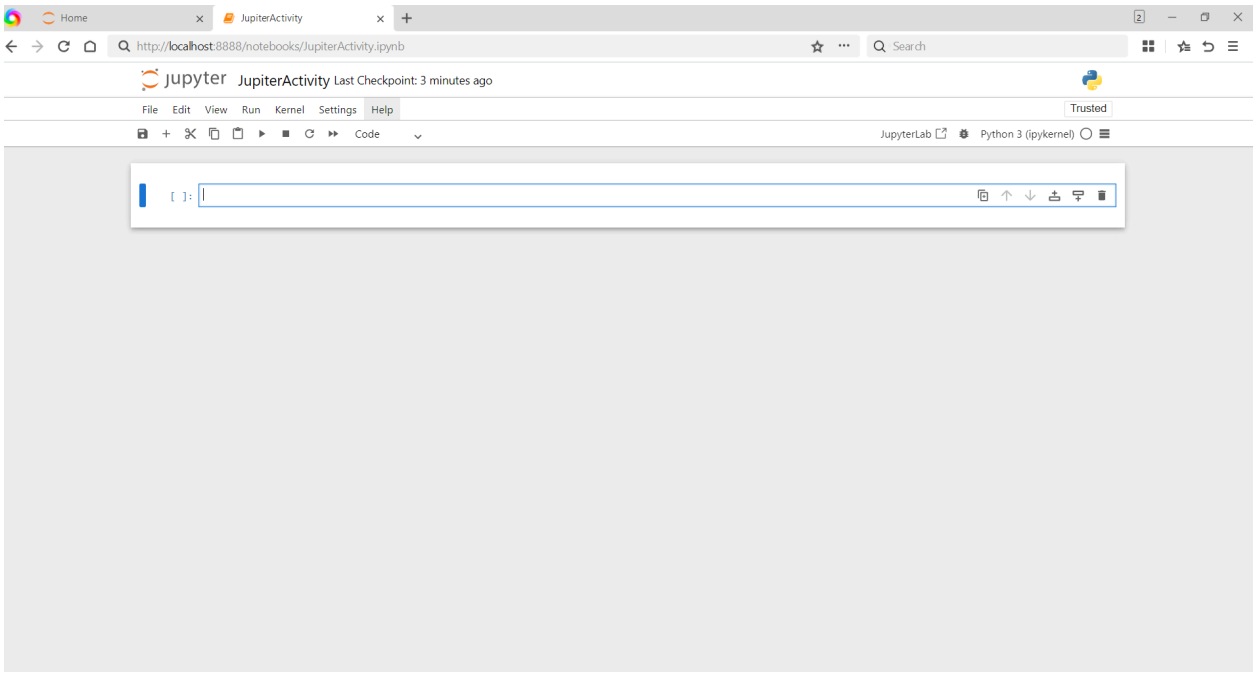
Lagunero, Jeffrey N.	Prof:Eliezer Refill
IT32S2	Data Mining and Warehousing

CASE 1 | Working with CSV Files in Jupyter Notebook

Instructions:

Part 1: Creating a CSV File

1. Open Jupyter Notebook and create a new Python notebook.



2. Run the following code to create a **students.csv** file with sample data

```
[1]: import csv

# Sample student data
students_data = [
    ["ID", "Name", "Age", "Course"],
    [1, "Jeff", 20, "Computer Science"],
    [2, "Cj", 22, "Information Technology"],
    [3, "Carl", 27, "Data Science"]
]

# Create and write data to the CSV file
with open("students.csv", mode="w", newline="") as file:
    writer = csv.writer(file)
    writer.writerows(students_data)

print("CSV file 'students.csv' has been created successfully!")
```

```
CSV file 'students.csv' has been created successfully!
```

Part 2: Locating the CSV File

3. Run the following code to **check the current directory** where Jupyter Notebook is running:

```
: import os
print("Current Directory:", os.getcwd())
```

```
Current Directory: C:\Users\lagun
```

```
:
```

4. Run the following code to **find the exact path of students.csv**:

```
]: file_path = os.path.abspath("students.csv")
print("Full Path of students.csv:", file_path)
```

```
Full Path of students.csv: C:\Users\lagun\students.csv
```

Part 3: Loading and Displaying CSV Data

5. Load the **students.csv** file into Pandas and display its contents:

```
]: import pandas as pd

# Load students.csv into a DataFrame
df = pd.read_csv("students.csv")

# Display the contents
print(df)
```

```
   ID  Name  Age                   Course
0   1  Jeff   20        Computer Science
1   2    Cj   22  Information Technology
2   3  Carl   27            Data Science
```

Part 4: Modifying the CSV File

6. Add a new column **"Course"** and save the changes:

```
[5]: import csv  # Import the csv module

new_student = [5, "Kent", 46, "Software Engineering"]

# Append new student to CSV file
with open("students.csv", mode="a", newline="") as file:
    writer = csv.writer(file)  # Indented correctly
    writer.writerow(new_student)  # Indented correctly

print("New student record added successfully!")  # Not inside the with block
```

```
New student record added successfully!
```

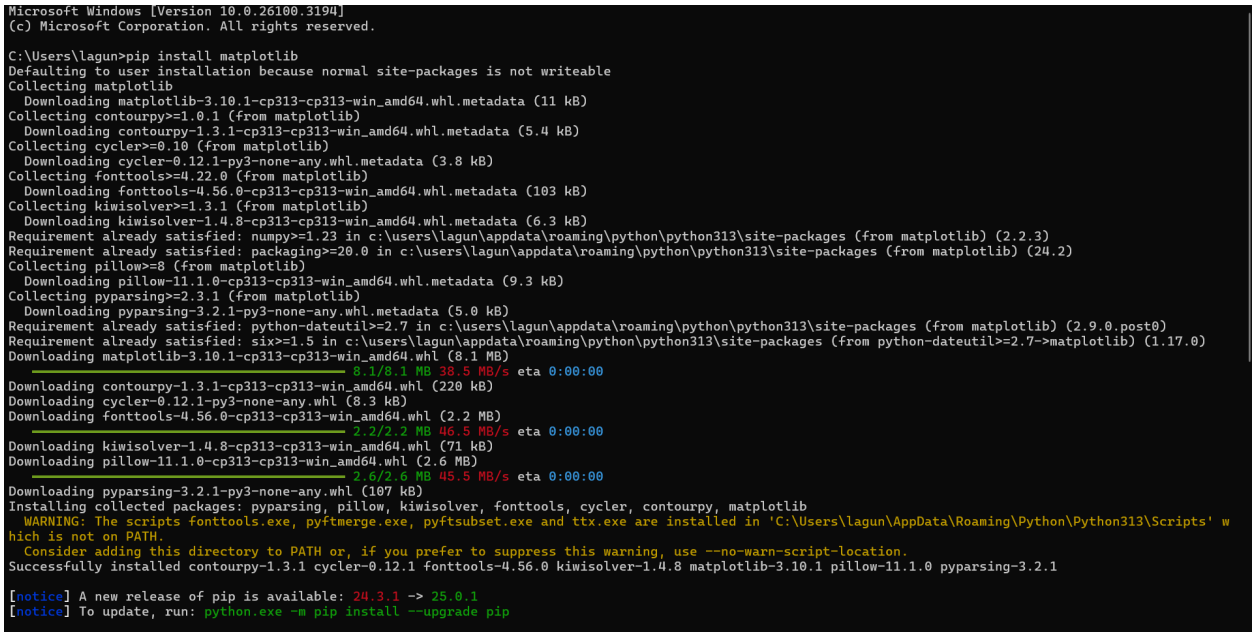
```
[6]: df = pd.read_csv("students.csv")
print(df)
```

```
   ID   Name  Age                   Course
0   1   Jeff   20        Computer Science
1   2     Cj   22  Information Technology
2   3   Carl   27            Data Science
3   4  David   23    Software Engineering
4   5   Kent   46    Software Engineering
```

CASE 2 | Introduction to Pandas and Matplotlib in Jupyter Notebook

Part 1: Installing and Importing Pandas & Matplotlib

1. Open Jupyter Notebook and create a new Python notebook.
2. Install Pandas and Matplotlib (if not already installed) by running:



Part 2: Creating and Loading a CSV File

4. Create a sample CSV file called **students.csv**:

```python
import csv

# Sample student data
data = [
    ["ID", "Name", "Age", "Score"],
    [1, "Stan", 21, 85],
    [2, "Cj", 22, 78],
    [3, "Carl", 20, 92],
    [4, "Abel", 23, 88],
    [5, "Jeff", 21, 76]
]

# Write data to students.csv
with open("students.csv", mode="w", newline="") as file:
    writer = csv.writer(file)
    writer.writerows(data)

print("CSV file 'students.csv' created successfully!")
```

```
CSV file 'students.csv' created successfully!
```

```python
import pandas as pd  # Ensure pandas is imported

df = pd.read_csv("students.csv")
print(df)
```

```
   ID  Name  Age  Score
0   1  Stan   21     85
1   2    Cj   22     78
2   3  Carl   20     92
3   4  Abel   23     88
4   5  Jeff   21     76
```

Part 3: Analyzing Data with Pandas

6. Display basic information about the dataset:

```
print("Dataset Information:")
print(df.info())  # Shows column details and data types

print("\nBasic Statistics:")
print(df.describe())  # Provides summary statistics (only numerical columns)
```

```
Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      5 non-null      int64
 1   Name    5 non-null      object
 2   Age     5 non-null      int64
 3   Score   5 non-null      int64
dtypes: int64(3), object(1)
memory usage: 292.0+ bytes
None

Basic Statistics:
             ID        Age      Score
count  5.000000   5.000000   5.000000
mean   3.000000  21.400000  83.800000
std    1.581139   1.140175   6.723095
min    1.000000  20.000000  76.000000
25%    2.000000  21.000000  78.000000
50%    3.000000  21.000000  85.000000
75%    4.000000  22.000000  88.000000
max    5.000000  23.000000  92.000000
```

7. Filter students who scored above 80:

```
high_scorers = df[df["Score"] > 80]
print("\nStudents who scored above 80:")
print(high_scorers)
```

```
Students who scored above 80:
   ID  Name  Age  Score
0   1  Stan   21     85
2   3  Carl   20     92
3   4  Abel   23     88
```

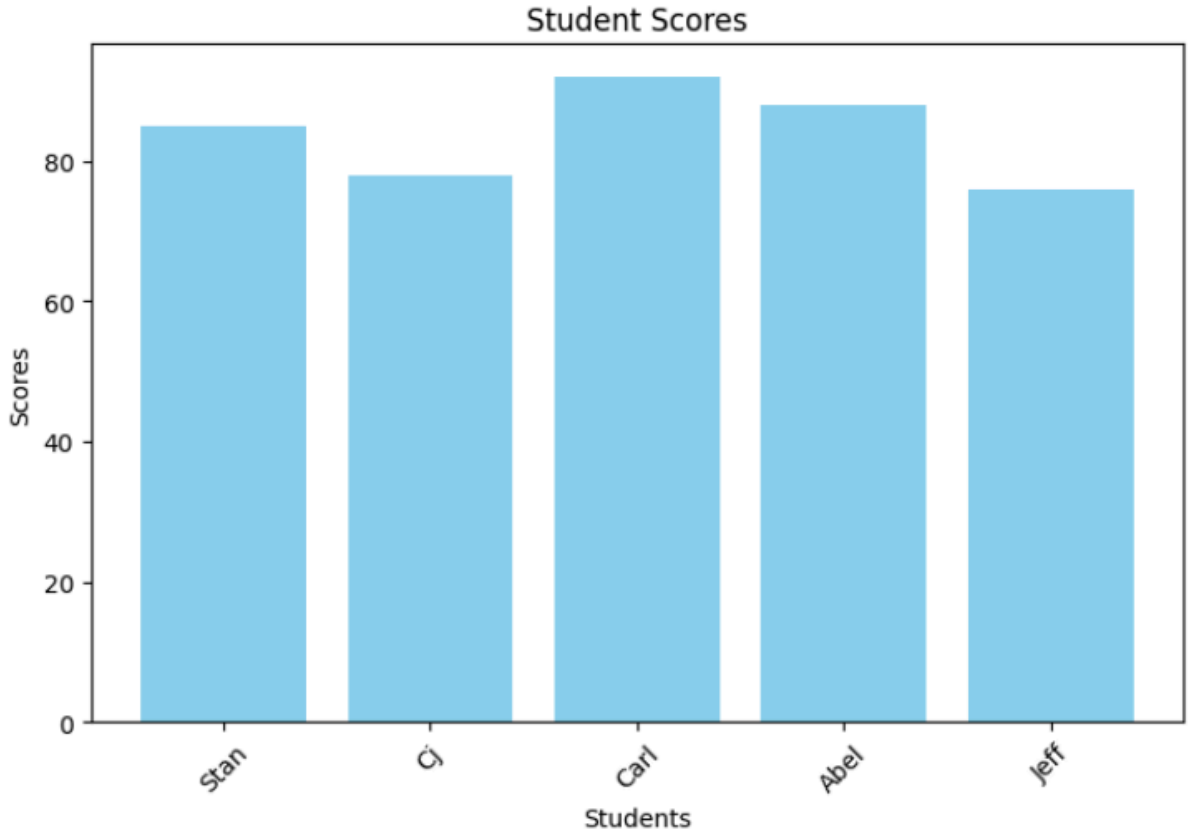
Part 4: Visualizing Data with Matplotlib

8. Create a **bar chart** to show students' grades:

```
import matplotlib.pyplot as plt  # Ensure matplotlib is imported

plt.figure(figsize=(8, 5))
plt.bar(df["Name"], df["Score"], color='skyblue')
plt.xlabel("Students")
plt.ylabel("Scores")
plt.title("Student Scores")
plt.xticks(rotation=45)  # Rotate names for better visibility
plt.show()
```

```
Matplotlib is building the font cache; this may take a moment.
```



9. Create a **pie chart** to visualize the age distribution:

```
import matplotlib.pyplot as plt  # Ensure matplotlib is imported
plt.figure(figsize=(6, 6))
df["Age"].value_counts().plot.pie(autopct="%1.1f%%", colors=["lightcoral", "gold", "lightblue", "lightgreen"])
plt.title("Age Distribution of Students")
plt.ylabel("")  # Hide the y-label
plt.show()
```