Experiment 1

Objective:

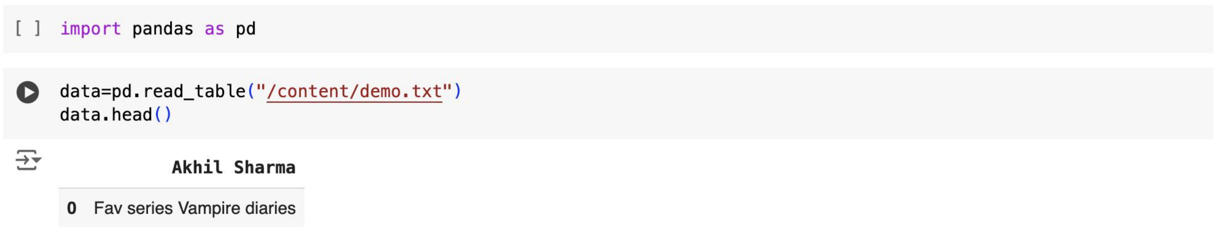
Reading Different Types of Data Sets (.txt, .csv) from Web and Disk and Writing in File in Specific Disk Location

Tools & Libraries:

- **Python:** A high-level programming language used for general-purpose programming.
- **pandas:** A powerful data manipulation and analysis library for Python. It provides data structures and functions needed to manipulate structured data seamlessly.

Procedure and Explanation:

1. **Reading .txt File from Disk:**



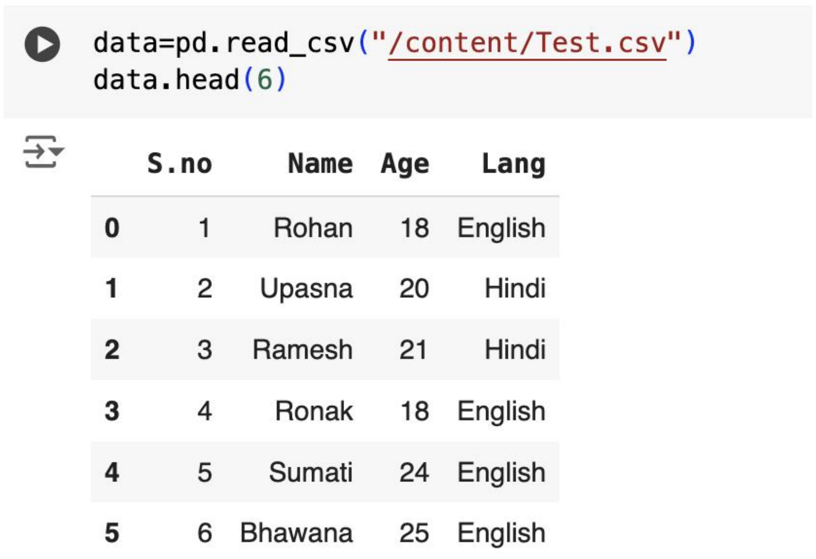
```
import pandas as pd
```

```
data=pd.read_table("/content/demo.txt")
data.head()
```

```
            Akhil Sharma
0  Fav series Vampire diaries
```

   - ***open():*** Opens the file in read mode.
   - ***file.read():*** Reads the content of the file.
   - ***print(data):*** Prints the content of the file.

2. **Reading .csv File from Disk:**

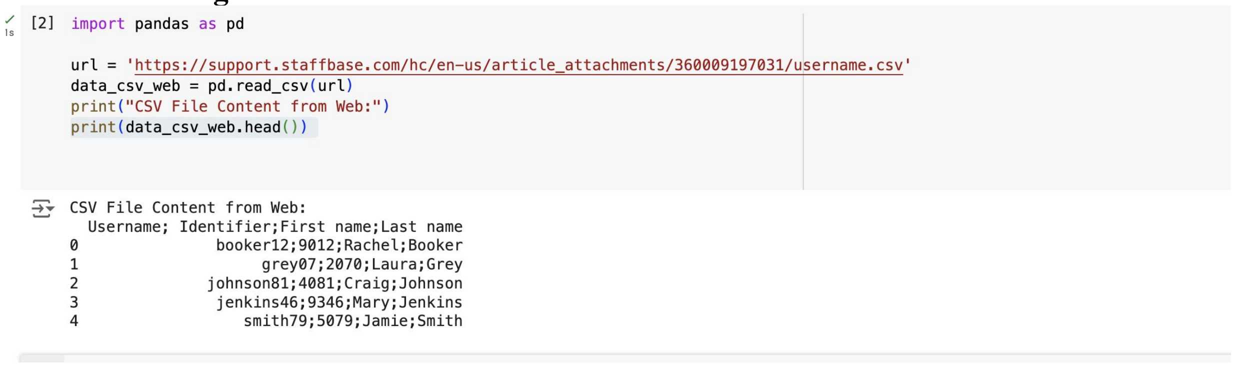


```
data=pd.read_csv("/content/Test.csv")
data.head(6)
```

```
   S.no     Name  Age     Lang
0     1    Rohan   18  English
1     2   Upasna   20    Hindi
2     3   Ramesh   21    Hindi
3     4    Ronak   18  English
4     5   Sumati   24  English
5     6  Bhawana   25  English
```

   - ***pd.read_csv():*** Reads a CSV file into a DataFrame.
   - ***data.head():*** Displays the first few rows of the DataFrame.

3. Reading .csv File from Web:



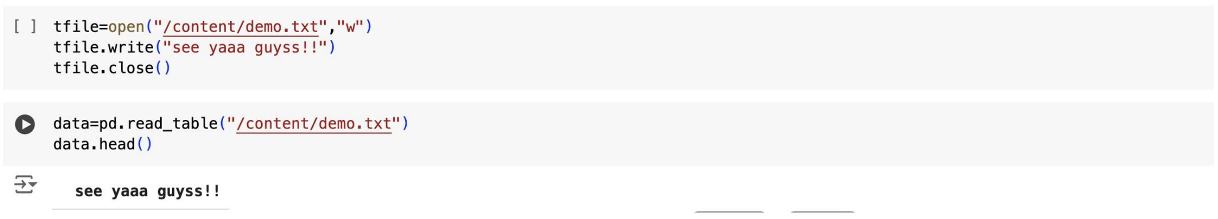
```
[2] import pandas as pd

    url = 'https://support.staffbase.com/hc/en-us/article_attachments/360009197031/username.csv'
    data_csv_web = pd.read_csv(url)
    print("CSV File Content from Web:")
    print(data_csv_web.head())
```

```
CSV File Content from Web:
  Username; Identifier;First name;Last name
0               booker12;9012;Rachel;Booker
1                    grey07;2070;Laura;Grey
2              johnson81;4081;Craig;Johnson
3               jenkins46;9346;Mary;Jenkins
4                  smith79;5079;Jamie;Smith
```

- ***pd.read_csv():*** Reads a CSV file into a DataFrame.
- ***data.head():*** Displays the first few rows of the DataFrame.

4. Writing Data to a .txt File:



```
[ ] tfile=open("/content/demo.txt","w")
    tfile.write("see yaaa guyss!!")
    tfile.close()
```

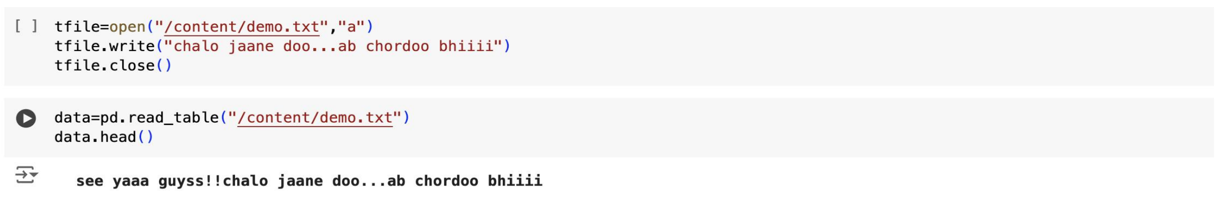
```
data=pd.read_table("/content/demo.txt")
data.head()
```

```
see yaaa guyss!!
```

- ***open():*** Opens the file in write mode.
- ***file.write():*** Writes the DataFrame content to a text file.

❖ ***When you open a file in write mode ('w'), it truncates the file to zero length, effectively deleting any existing content and then writing the new data. If you want to append new data to the existing content, you should open the file in append mode ('a').***

5. Appending on Data to a .txt File:



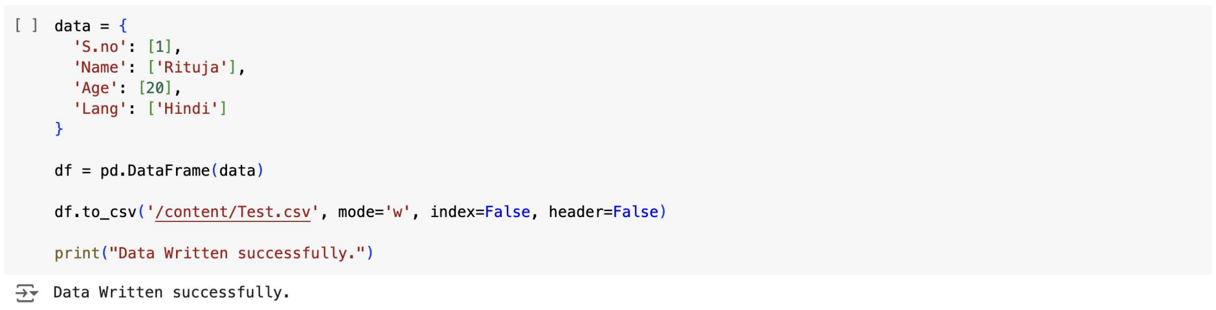
```
[ ] tfile=open("/content/demo.txt","a")
    tfile.write("chalo jaane doo...ab chordoo bhiiii")
    tfile.close()
```

```
data=pd.read_table("/content/demo.txt")
data.head()
```

```
see yaaa guyss!!chalo jaane doo...ab chordoo bhiiii
```

- Defines file_path as a raw string to ensure that backslashes are treated as literal characters.
- Uses a with statement to open the file in append mode, ensuring that the file is properly closed after the block is executed.
- Writes the specified string to the file.
- Automatically closes the file at the end of the with block, ensuring no need to explicitly close the file.

6.Writing Data to a .csv File:



```
data = {
  'S.no': [1],
  'Name': ['Rituja'],
  'Age': [20],
  'Lang': ['Hindi']
}

df = pd.DataFrame(data)

df.to_csv('/content/Test.csv', mode='w', index=False, header=False)

print("Data Written successfully.")
```

```
Data Written successfully.
```

1. Importing the pandas Library:
The `import pandas as pd` statement loads the pandas library, which provides data structures and data analysis tools.

2. Defining the Data:
The `data` dictionary is structured with keys representing column names and values as lists of column data. This dictionary serves as the source for the DataFrame.

3. Creating a DataFrame:
`pd.DataFrame(data)` converts the dictionary into a pandas DataFrame. This DataFrame organizes the data into a tabular structure with labeled axes.

4. Writing to CSV:
The `df.to_csv('/content/Test.csv', mode='w', index=False, header=False)` method writes the DataFrame to a CSV file. The `mode='w'` parameter specifies that the file should be overwritten if it already exists. The `index=False` parameter ensures that row indices are not included in the file, and `header=False` excludes column headers from the output.

5. Output Confirmation:
The `print("Data written successfully.")` statement provides a confirmation message indicating that the data has been successfully written to the specified file.

This code succinctly demonstrates data handling and persistence using pandas, which is a common practice in data processing and analysis workflows.

- ***data.to_csv():*** Writes the DataFrame content to a CSV file.
- ***index=False:*** Prevents writing row indices to the file