Date 28/9/25
MON TUE WED THU FRI SAT SUN

Module 4:

Assignment Questions:

Q1 What is Pandas in Python? How do you import the Pandas library in python? write the correct syntax

⇒ Pandas is a powerful Python library used for data manipulation and data analysis.

- It provides two main data structures: Series (1-dimensional like a) and dataframe (2-dimensional, like a table).
- With pandas, you can handle structured data (like Excel sheets, CSV files, databases) easily.

Import syntax: import pandas as pd.

* Here, "pd" is an alias commonly used for Pandas.

Q2 Explain pandas importance in database handling and data analysis.

- Pandas is very important in database handling and data analysis because:

Date : __/__/____
MON TUE WED THU FRI SAT SUN

1. It can read data from CSV, Excel, SQL databases, JSON, HTML, etc.

2. It provides easy-to-use data structures (Series and Dataframe) for tabular data.

3. You can filter, sort, group, and merge data quickly.

4. It has built-in functions for statistics, cleaning, and reshaping data.

5. It integrates well with Numpy, Matplotlib, and Scikit-learn.

Thus, Pandas makes handling large amounts of data simpler, faster, and more efficient than using lists or dictionaries.

Q3 Define a Series in Pandas. How is it different from a Python list? Give an example.

⇒ • A Series in Pandas is a one-dimensional labeled array that can hold any type of data (integers, strings, floats, etc.).

• Difference from Python list :
  • A list is only a sequence of elements without labels.

Date : __/__/____
MON TUE WED THU FRI SAT SUN

- A series has both values and indexes (labels).

Example:
```
import pandas as pd
data = [10, 20, 30]
series = pd.Series(data)
print(series)
```

output:
```
0    10
1    20
2    30
dtype: int64
```

Here, [0, 1, 2] are indexes which lists do not provide by default.

Q4 Explain in detail the difference between Series and DataFrame with suitable code examples.

Answer:

- Series: One-dimensional labeled data (like a single column).
- Dataframe: Two-dimensional table (like rows & columns in Excel).

Example of series:-
```
import pandas as pd
s = pd.Series([10, 20, 30])
print(s)
```

Date : __/__/____
MON TUE WED THU FRI SAT SUN

Example of Dataframe:

```
import pandas as pd

data = {
    "Name": ["Ajay", "Ravi", "Neha"],
    "Marks": [85, 90, 95]
    }
df = pd.DataFrame(data)
print(df)
```

Output:

	Name	Marks
0	Ajay	85
1	Ravi	90
2	Neha	95

Conclusion:
- series → single column
- Dataframe → Multiple columns (Table).

Q5. Explain the role of Pandas in data analysis with Python. How does it simplify working with large datasets compared to traditional Python lists and dictionaries?

⇒ • Pandas provides powerful tools for analyzing large datasets.

• It is faster and more efficient than lists and dictionaries because:

1. You can directly filter rows / colomns with conditions.

Date : __/__/____
MON TUE WED THU FRI SAT SUN

2. It has built-in mathematical and statistical functions.

3. You can read/write large CSV or Excel files in just one line.

4. Dataframe works like an Excel sheet inside python.

Example :
Using a list & dictionary is slow:

Code ⇒
```
students = [{"Name": "Ajay", "Marks": 85},
            {"Name": "Ravi", "Marks": 90}]
```

But with pandas Dataframe :

Code⇒
```
import pandas as pd
data = {"Name": ["Ajay", "Ravi"],
        "Marks": [85, 90]}

df = pd.Dataframe(data)
print(df)
```

** Output is a clean table that is easy to work with **

Q6 → Consider a scenario, where a dataset is given as

Date : __/__/____
MON TUE WED THU FRI SAT SUN

code→
```
data = {
  "Year" : [2005, 2006, 2007, 2008],
  "Sales" : [500, 400, 450, 750]
    }
```

which is the sales report of the 4 years. Then to display the data frame row-wise, you have got the error "ValueError: 5 is not in range"

Deduce the possible reason for the error and write the corrected code

Hint: Check the correct location of the row.

⇒ • This error occurs because you are toying to access a row index that does not exist.

• The dataset has 4 rows (0-3 indexes), so if you try to access row index 5, you get this error.

Correct code :-

```
import pandas as pd
data = {
  "Year" : [2005, 2006, 2007, 2008],
  "Sales" : [500, 400, 450, 750]
    }
```

```
df = pd.DataFrame(data)
print(df.loc[0])    # first row
print(df.loc[3])    # last row
```

Q7. Consider a dataset containing more than 1000 records in a CSV file. Read the content and save it to a Dataframe. Measure the number of rows actually present in the dataset.

⇒ We can use len() with Pandas to count rows.

Code:

```
import pandas as pd
df = pd.read_csv("data.csv")
print("Number of rows:", len(df))
```

We can also set display options for pandas:

code:

```
pd.options.display.max_rows = 1000
```

This ensures Pandas shows all rows if required.

Q8. ~~Cop~~ Suppose you have two different CSV files and you have to merge the content horizontally.

Date : __/__/____
MON TUE WED THU FRI SAT SUN

⇒ Code:

```
import pandas as pd

df1 = pd.read_csv ("file1.csv")
df2 = pd.read_csv ("file2.csv")

merged = pd.concat ([df1, df2], axis = 1)
print (merged)
```

★⇒ This joins them column by column (horizontally).

Q9. CSV file "students.csv" contains columns : Name, Section, Marks, Gender. Categorize it into two files : Male and Female.

⇒ Code :

```
import pandas as pd

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

male_students = df [df ['Gender'] == 'Male']

female_students = df [df ['Gender'] == 'Female']

male_students.to_csv ("male_students.csv", index = False)

female_students.to_csv ("female_students_csv", index = False)
```

Date : __/__/____
MON TUE WED THU FRI SAT SUN

This saves two separate CSV files.

Q10. Consider a dataset :

"x" : [5, 2, 7, 0],
"y" : [4, 7, 5, 1],
"z" : [9, 3, 5, 1]

Measure the correlation among first two columns.

⇒ Code:

```
import pandas as pd
data = {
  "x" : [5, 2, 7, 0],
  "y" : [4, 7, 5, 1],
  "z" : [9, 3, 5, 1]
  }

df = pd.DataFrame (data)
print (df [["x", "y"]].corr ())
```

* This gives the correlation coefficient between "x" and "y".

Q11. Write a program to compose a dictionary which contains English Alphabets both uppercase and lowercase and their ASCII values.

Date : __/__/____
MON TUE WED THU FRI SAT SUN

⇒ Code :

```
import json

ascii_dict = {}

for ch in range(65, 91):        # A-Z
    ascii_dict[chr(ch)] = ch

for ch in range(97, 123):       # a-z
    ascii_dict[chr(ch)] = ch

print(json.dumps(ascii_dict, indent=4))
```

* This prints a dictionary with all alphabets and their ASCII values.

* A - American
  S - Standard
  C - Code for
  I - Information
  I - Interchange.