	Marwadi University **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

Aim: Practical based on Pandas Data Structures

IDE:

What is Python Pandas?

Pandas is a powerful, open-source data analysis and manipulation package for Python. It provides data structures and functions needed to work on structured data seamlessly and efficiently.

What Is Pandas Used For?

Pandas is extensively used for:

- Data Cleaning: Handling missing values, duplications, and incorrect data formats.
- Data Manipulation: Filtering, transforming, and merging datasets.
- Data Analysis: Performing statistical analysis and aggregations.
- Data Visualization: Creating plots and charts to visualize data trends and patterns.
- Time Series Analysis: Handling and manipulating time series data.

Run the following command to install Pandas:

pip install pandas

import pandas as pd

print(pd.__version__)

Pandas Series

A Pandas Series is a one-dimensional labeled array capable of holding any data type. It is similar to a column in a spreadsheet or a SQL table.

Example:

```
import pandas as pd
# Creating a Series
data = [1, 2, 3, 4, 5]
series = pd.Series(data)
print(series)
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

Output:

```
In [1]:
   ...: import pandas as pd
   ...: data = [1, 2, 3, 4, 5]
   ...: series = pd.Series(data)
   ...: print(series)
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

Basic Operations on Series

Perform various operations on Series, such as arithmetic operations, filtering, and statistical calculations.

Example:

```
# Arithmetic Operations
series2 = series + 10
print(series2)
# Filtering
filtered_series = series[series > 2]
print(filtered_series)
# Statistical Calculations
mean_value = series.mean()
print(mean_value)
```

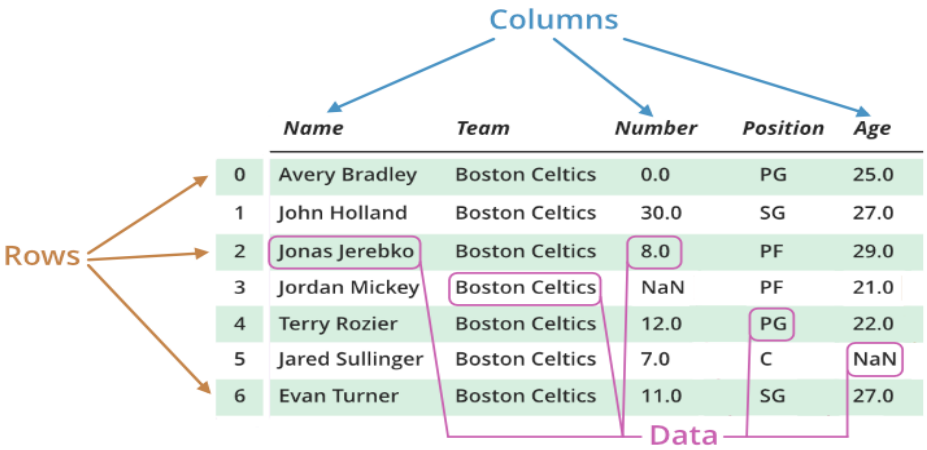
Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
In [2]: series2 = series + 10
   ...: print(series2)
   ...: filtered_series = series[series > 2]
   ...: print(filtered_series)
   ...: mean_value = series.mean()
   ...: print(mean_value)
0    11
1    12
2    13
3    14
4    15
dtype: int64
2    3
3    4
4    5
dtype: int64
3.0
```

Pandas Dataframe

Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the data, rows, and columns.



	Name	Team	Number	Position	Age
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
# Creating a DataFrame
data = {
   'Name': ['Alice', 'Bob', 'Charlie'],
   'Age': [25, 30, 35],
   'City': ['New York', 'Los Angeles', 'Chicago']
}
df = pd.DataFrame(data)
print(df)
```

Output :

```
In [3]: data = {
   ...: 'Name': ['Alice', 'Bob', 'Charlie'],
   ...: 'Age': [25, 30, 35],
   ...: 'City': ['New York', 'Los Angeles', 'Chicago']
   ...: }
   ...: df = pd.DataFrame(data)
   ...: print(df)
   ...:
      Name  Age         City
0    Alice   25     New York
1      Bob   30  Los Angeles
2  Charlie   35      Chicago
```

Basic Operations on Dataframes

DataFrames support a wide range of operations for data manipulation and analysis.

```
# Accessing Columns (# select one column)
print(df[['Name']])
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

Output:

```
In [4]: print(df[['Name']])
      Name
0    Alice
1      Bob
2  Charlie
```

Adding a New Column
df['Salary'] = [70000, 80000, 90000]
print(df)
Output:

```
In [5]: df['Salary'] = [70000, 80000, 90000]
   ...: print(df)
      Name  Age         City  Salary
0    Alice   25     New York   70000
1      Bob   30  Los Angeles   80000
2  Charlie   35      Chicago   90000
```

# Dropping a Column
df = df.drop('City', axis=1)
print(df)
Output:

```
In [6]: df = df.drop('City', axis=1)
   ...: print(df)
   ...:
      Name  Age  Salary
0    Alice   25   70000
1      Bob   30   80000
2  Charlie   35   90000
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

The DataFrame is like a table with rows and columns.
Pandas use the loc attribute to return one or more specified row(s)

```
# Return row 0:
print(df.loc[[0]])
```

Output:

```
In [7]: print(df.loc[[0]])
    Name  Age  Salary
0  Alice   25   70000
```

```
#Return row 0 and 1:
#use a list of indexes:
print(df.loc[[0, 1]])
```

Output:

```
In [8]: print(df.loc[[0, 1]])
    Name  Age  Salary
0  Alice   25   70000
1    Bob   30   80000
```

Named Indexes

With the index argument, you can name your own indexes.
Example:
Add a list of names to give each row a name:

```
import pandas as pd
data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}
df = pd.DataFrame(data, index = ["day1", "day2", "day3"])
print(df)
```

Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
In [9]: data = {
   ...: "calories": [420, 380, 390],
   ...: "duration": [50, 40, 45]
   ...: }
   ...: df = pd.DataFrame(data, index = ["day1", "day2", "day3"])
   ...: print(df)
      calories  duration
day1       420        50
day2       380        40
day3       390        45
```

Explanation of Key Pandas Functions
Reading and Writing Data:

Reading Data: Read a CSV file into a DataFrame.
Example:
dat = pd.read_csv("data.csv")
print(dat)
Output:

```
   ...: print(dat)
   data
0  1234
```

Writing Data: Write a DataFrame to a CSV file.
Note: Other Ways to Save Pandas DataFrames (to_excel(), to_json(), to_hdf(), to_sql(), to_pickle())

Example:

```
Biodata = {'Name': ['John', 'Emily', 'Mike', 'Lisa'],
      'Age': [28, 23, 35, 31],
      'Gender': ['M', 'F', 'M', 'F']
      }
df = pd.DataFrame(Biodata)
# Save the dataframe to a CSV file
```

df.to_csv('Biodata.csv', index=False)
Output:

```
In [5]: Biodata = {'Name': ['John', 'Emily', 'Mike', 'Lisa'],
   ...:            'Age': [28, 23, 35, 31],
   ...:            'Gender': ['M', 'F', 'M', 'F']
   ...:           }
   ...: df = pd.DataFrame(Biodata)
   ...: df.to_csv('Biodata.csv', index=False)
   ...:
```

Data Inspection:

`df.head()`: Display the first few rows of the DataFrame.
`df.tail()`: Display the last few rows of the DataFrame.
`df.info()`: Display a summary of the DataFrame.
`df.describe()`: Provide descriptive statistics for numerical columns. (count: the number of non-null entries, mean: the mean value, std: the standard deviation, min: the minimum value, 25%, 50%, 75%: the lower, median, and upper quartiles, max: the maximum value)

Example:

```
dat = pd.read_csv("data.csv")
print(dat.info())
# shows first and last five rows
print(dat.head())
print(dat.tail())
print(dat.describe())
```

Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
In [6]: at = pd.read_csv("data.csv")
   ...: print(dat.info())
   ...: print(dat.head())
   ...: print(dat.tail())
   ...: print(dat.describe())
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1 entries, 0 to 0
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   data    1 non-null      int64
dtypes: int64(1)
memory usage: 140.0 bytes
None
   data
0  1234
   data
0  1234
          data
count     1.0
mean   1234.0
std       NaN
min    1234.0
25%    1234.0
50%    1234.0
75%    1234.0
max    1234.0
```

Data Selection and Indexing:

`dat[['A']]`: Select a column.

`dat[['A', 'B']]`: Select multiple columns.

`dat.loc[[0]]`: Select a row by label.

Example:

```
print(dat[['Name']])
print(dat[['Name','Number']])
print(dat.loc[[1]])
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

Output:

```
print(dat[['Name']])
print(dat[['Name','Number']])
print(dat.loc[[1]])
```

```
  Name City  Number
1    B    N       4
```

Data Manipulation:

dat['A'] = dat['A'] * 2: Modify a column.

dat['F'] = dat['A'] + dat['B']: Create a new column based on existing columns.

dat.drop(columns=['A']): Drop a column.

dat.drop(index=[0]): Drop a row.

Task

Create a DataFrame with 5 numeric columns

```
data = {
    'A': [np.nan, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'B': np.random.normal(50, 15, 10),
    'C': np.random.rand(10) * 100,
    'D': np.linspace(1, 10, 10),
    'E': np.logspace(1, 2, 10)
}
df = pd.DataFrame(data)
```

Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
data = {
    'A': [np.nan, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'B': np.random.normal(50, 15, 10),
    'C': np.random.rand(10) * 100,
    'D': np.linspace(1, 10, 10),
    'E': np.logspace(1, 2, 10)
}
df = pd.DataFrame(data)
print(df)
```

```
     A          B          C     D           E
0  NaN  43.693217  89.099029   1.0   10.000000
1  2.0  34.042308  30.303802   2.0   12.915497
2  3.0  13.149508  56.841348   3.0   16.681005
3  4.0  29.976347  49.594907   4.0   21.544347
4  5.0  70.917927  56.637306   5.0   27.825594
5  6.0  57.288888  91.098210   6.0   35.938137
6  7.0  58.104204  23.280560   7.0   46.415888
7  8.0  47.652953  92.271397   8.0   59.948425
8  9.0  45.472237  27.995592   9.0   77.426368
9 10.0  61.182469  47.305318  10.0  100.000000
```

Post Lab Exercise:

a. Write a Pandas program to add, subtract, multiple and divide two Pandas Series.

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```python
import pandas as pd
s1 = pd.Series([10, 20, 30, 40, 50])
s2 = pd.Series([2, 4, 6, 8, 10])

print("Series 1:")
print(s1)

print("\nSeries 2:")
print(s2)

print("\nAddition of two Series:")
print(s1 + s2)

print("\nSubtraction of two Series:")
print(s1 - s2)

print("\nMultiplication of two Series:")
print(s1 * s2)

print("\nDivision of two Series:")
print(s1 / s2)
```

Output:

	Marwadi University **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
Series 1:
0    10
1    20
2    30
3    40
4    50
dtype: int64

Series 2:
0     2
1     4
2     6
3     8
4    10
dtype: int64

Addition of two Series:
0    12
1    24
2    36
3    48
```

```
Subtraction of two Series:
0     8
1    16
2    24
3    32
4    40
dtype: int64

Multiplication of two Series:
0     20
1     80
2    180
3    320
4    500
dtype: int64

Division of two Series:
0    5.0
1    5.0
2    5.0
3    5.0
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

b. Write a Pandas program to convert a dictionary to a Pandas series.

```
data = {'a': 100, 'b': 200, 'c': 300, 'd': 400}
print("Dictionary:")
print(data)
series = pd.Series(data)
print("\nConverted Pandas Series:")
print(series)
```

Output:

```
In [2]: import pandas as pd
   ...: data = {'a': 100, 'b': 200, 'c': 300, 'd': 400}
   ...: print("Dictionary:")
   ...: print(data)
   ...: series = pd.Series(data)
   ...: print("\nConverted Pandas Series:")
   ...: print(series)
Dictionary:
{'a': 100, 'b': 200, 'c': 300, 'd': 400}

Converted Pandas Series:
a    100
b    200
c    300
d    400
dtype: int64
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

c. Write a Pandas program to create a series from a list, numpy array and dict.

```
import pandas as pd
import numpy as np
list_data = [10, 20, 30, 40]
series_from_list = pd.Series(list_data)
print("Series from list:")
print(series_from_list)

array_data = np.array([100, 200, 300, 400])
series_from_array = pd.Series(array_data)
print("\nSeries from NumPy array:")
print(series_from_array)

dict_data = {'a': 1, 'b': 2, 'c': 3, 'd': 4}
series_from_dict = pd.Series(dict_data)
print("\nSeries from dictionary:")
print(series_from_dict)
```

Output:

```
In [3]: import pandas as pd
   ...: import numpy as np
   ...: list_data = [10, 20, 30, 40]
   ...: series_from_list = pd.Series(list_data)
   ...: print("Series from list:")
   ...: print(series_from_list)
   ...:
   ...: array_data = np.array([100, 200, 300, 400])
   ...: series_from_array = pd.Series(array_data)
   ...: print("\nSeries from NumPy array:")
   ...: print(series_from_array)
   ...:
   ...: dict_data = {'a': 1, 'b': 2, 'c': 3, 'd': 4}
   ...: series_from_dict = pd.Series(dict_data)
   ...: print("\nSeries from dictionary:")
   ...: print(series_from_dict)
Series from list:
0    10
1    20
2    30
3    40
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

```
2    30
3    40
dtype: int64

Series from NumPy array:
0    100
1    200
2    300
3    400
dtype: int64

Series from dictionary:
a    1
b    2
c    3
d    4
dtype: int64
```

d. Write a Pandas program to stack two series vertically and horizontally

```
s1 = pd.Series([1, 2, 3, 4])
s2 = pd.Series([5, 6, 7, 8])
vertical = pd.concat([s1, s2])
print("Vertical Stacking:\n", vertical)
horizontal = pd.concat([s1, s2], axis=1)
print("\nHorizontal Stacking:\n", horizontal)
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on Pandas Data Structures	
Experiment No: 09	**Date:**	**Enrollment No: 92400133110**

Output:

```
In [4]: s1 = pd.Series([1, 2, 3, 4])
   ...: s2 = pd.Series([5, 6, 7, 8])
   ...: vertical = pd.concat([s1, s2])
   ...: print("Vertical Stacking:\n", vertical)
   ...: horizontal = pd.concat([s1, s2], axis=1)
   ...: print("\nHorizontal Stacking:\n", horizontal)
Vertical Stacking:
 0    1
1    2
2    3
3    4
0    5
1    6
2    7
3    8
dtype: int64
```

```
Vertical Stacking:
 0    1
1    2
2    3
3    4
0    5
1    6
2    7
3    8
dtype: int64

Horizontal Stacking:
    0  1
0  1  5
1  2  6
2  3  7
3  4  8
```

Github:
https://github.com/hemanth-singampalli/pwp.git