**Pandas Lab Experiments**

**11. Pandas Data Series:**

**1. Write a Pandas program to create and display a one-dimensional array-like object containing an array of data using Pandas module.**

**Python Code :**

import pandas as pd

ds = pd.Series([2, 4, 6, 8, 10])

print(ds)

**Output:**

0 2

1 4

2 6

3 8

4 10

dtype: int64

**2. Write a Pandas program to convert a Panda module Series to Python list and it's type.**

**Python Code :**

import pandas as pd

ds = pd.Series([2, 4, 6, 8, 10])

print("Pandas Series and type")

print(ds)

print(type(ds))

print("Convert Pandas Series to Python list")

print(ds.tolist())

print(type(ds.tolist()))

**Output:**

Pandas Series and type

0 2

1 4

2 6

3 8

4 10

dtype: int64

<class 'pandas.core.series.Series'>

Convert Pandas Series to Python list

[2, 4, 6, 8, 10]

<class 'list'>

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

**Python Code:**

import pandas as pd

ds1 = pd.Series([2, 4, 6, 8, 10])

ds2 = pd.Series([1, 3, 5, 7, 9])

ds = ds1 + ds2

print("Add two Series:")

print(ds)

print("Subtract two Series:")

ds = ds1 - ds2

print(ds)

print("Multiply two Series:")

ds = ds1 \* ds2

print(ds)

print("Divide Series1 by Series2:")

ds = ds1 / ds2

print(ds)

**Output:**

Add two Series:

0 3

1 7

2 11

3 15

4 19

dtype: int64

Subtract two Series:

0 1

1 1

2 1

3 1

4 1

dtype: int64

Multiply two Series:

0 2

1 12

2 30

3 56

4 90

dtype: int64

Divide Series1 by Series2:

0 2.000000

1 1.333333

2 1.200000

3 1.142857

4 1.111111

dtype: float64

**4. Write a Pandas program to compare the elements of the two Pandas Series**

**Python Code :**

import pandas as pd

ds1 = pd.Series([2, 4, 6, 8, 10])

ds2 = pd.Series([1, 3, 5, 7, 10])

print("Series1:")

print(ds1)

print("Series2:")

print(ds2)

print("Compare the elements of the said Series:")

print("Equals:")

print(ds1 == ds2)

print("Greater than:")

print(ds1 > ds2)

print("Less than:")

print(ds1 < ds2)

**Output:**

Series1:

0 2

1 4

2 6

3 8

4 10

dtype: int64

Series2:

0 1

1 3

2 5

3 7

4 10

dtype: int64

Compare the elements of the said Series:

Equals:

0 False

1 False

2 False

3 False

4 True

dtype: bool

Greater than:

0 True

1 True

2 True

3 True

4 False

dtype: bool

Less than:

0 False

1 False

2 False

3 False

4 False

dtype: bool

**5. Write a Pandas program to convert a dictionary to a Pandas series.**

**Python Code :**

import pandas as pd

d1 = {'a': 100, 'b': 200, 'c':300, 'd':400, 'e':800}

print("Original dictionary:")

print(d1)

new\_series = pd.Series(d1)

print("Converted series:")

print(new\_series)

**Output:**

Original dictionary:

{'a': 100, 'b': 200, 'c': 300, 'd': 400, 'e': 800}

Converted series:

a 100

b 200

c 300

d 400

e 800

dtype: int64

**6. Write a Pandas program to convert a NumPy array to a Pandas series.**

**Python Code :**

import numpy as np

import pandas as pd

np\_array = np.array([10, 20, 30, 40, 50])

print("NumPy array:")

print(np\_array)

new\_series = pd.Series(np\_array)

print("Converted Pandas series:")

print(new\_series)

**Output:**

NumPy array:

[10 20 30 40 50]

Converted Pandas series:

0 10

1 20

2 30

3 40

4 50

dtype: int64

**12. Pandas Data Frames:**

**1. Write a Pandas program to create and display a DataFrame from a specified dictionary data which has the index labels.**

Sample Python dictionary data and list labels:  
exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],  
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],  
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],  
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}  
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

**Python Code :**

import pandas as pd

import numpy as np

exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam\_data , index=labels)

print(df)

**Output:**

name score attempts qualify

a Anastasia 12.5 1 yes

b Dima 9.0 3 no

c Katherine 16.5 2 yes

d James NaN 3 no

e Emily 9.0 2 no

f Michael 20.0 3 yes

g Matthew 14.5 1 yes

h Laura NaN 1 no

i Kevin 8.0 2 no

j Jonas 19.0 1 yes

**2. Write a Pandas program to display a summary of the basic information about a specified DataFrame and its data.**

**Python Code:**

import pandas as pd

import numpy as np

exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam\_data , index=labels)

print("Summary of the basic information about this DataFrame and its data:")

print(df.info())

**Output:**

Summary of the basic information about this DataFrame and its data:

<class 'pandas.core.frame.DataFrame'>

Index: 10 entries, a to j

Data columns (total 4 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 name 10 non-null object

1 score 8 non-null float64

2 attempts 10 non-null int64

3 qualify 10 non-null object

dtypes: float64(1), int64(1), object(2)

memory usage: 400.0+ bytes

None

**3. Write a Pandas program to change the name 'James' to 'Suresh' in name column of the DataFrame.**

**Python Code :**

import pandas as pd

import numpy as np

exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam\_data , index=labels)

print("Original rows:")

print(df)

print("\nChange the name 'James' to ‘Suresh’:")

df['name'] = df['name'].replace('James', 'Suresh')

print(df)

**Output:**

Original rows:

name score attempts qualify

a Anastasia 12.5 1 yes

b Dima 9.0 3 no

c Katherine 16.5 2 yes

d James NaN 3 no

e Emily 9.0 2 no

f Michael 20.0 3 yes

g Matthew 14.5 1 yes

h Laura NaN 1 no

i Kevin 8.0 2 no

j Jonas 19.0 1 yes

Change the name 'James' to ‘Suresh’:

name score attempts qualify

a Anastasia 12.5 1 yes

b Dima 9.0 3 no

c Katherine 16.5 2 yes

d Suresh NaN 3 no

e Emily 9.0 2 no

f Michael 20.0 3 yes

g Matthew 14.5 1 yes

h Laura NaN 1 no

i Kevin 8.0 2 no

j Jonas 19.0 1 yes

**4. Write a Pandas program to insert a new column in existing DataFrame.**

**Python Code :**

import pandas as pd

import numpy as np

exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam\_data , index=labels)

print("Original rows:")

print(df)

color = ['Red','Blue','Orange','Red','White','White','Blue','Green','Green','Red']

df['color'] = color

print("\nNew DataFrame after inserting the 'color' column")

print(df)

**Output:**

Original rows:

name score attempts qualify

a Anastasia 12.5 1 yes

b Dima 9.0 3 no

c Katherine 16.5 2 yes

d James NaN 3 no

e Emily 9.0 2 no

f Michael 20.0 3 yes

g Matthew 14.5 1 yes

h Laura NaN 1 no

i Kevin 8.0 2 no

j Jonas 19.0 1 yes

New DataFrame after inserting the 'color' column

name score attempts qualify color

a Anastasia 12.5 1 yes Red

b Dima 9.0 3 no Blue

c Katherine 16.5 2 yes Orange

d James NaN 3 no Red

e Emily 9.0 2 no White

f Michael 20.0 3 yes White

g Matthew 14.5 1 yes Blue

h Laura NaN 1 no Green

i Kevin 8.0 2 no Green

j Jonas 19.0 1 yes Red

**5. Write a Pandas program to get list from DataFrame column headers.**

**Python Code :**

import pandas as pd

import numpy as np

exam\_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam\_data , index=labels)

print(list(df.columns.values))

**Output:**

['name', 'score', 'attempts', 'qualify']

**13. Pandas Index:**

1. **Write a Pandas program to display the default index and set a column as an Index in a given dataframe.**

**Test Data:**

0 s001 V Alberto Franco 15/05/2002 35 street1 t1

1 s002 V Gino Mcneill 17/05/2002 32 street2 t2

2 s003 VI Ryan Parkes 16/02/1999 33 street3 t3

3 s001 VI Eesha Hinton 25/09/1998 30 street1 t4

4 s002 V Gino Mcneill 11/05/2002 31 street2 t5

5 s004 VI David Parkes 15/09/1997 32 street4 t6

**Python Code :**

import pandas as pd

df = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4'],

't\_id':['t1', 't2', 't3', 't4', 't5', 't6']})

print("Default Index:")

print(df.head(10))

print("\nschool\_code as new Index:")

df1 = df.set\_index('school\_code')

print(df1)

print("\nt\_id as new Index:")

df2 = df.set\_index('t\_id')

print(df2)

**Output:**

Default Index:

school\_code class name date\_Of\_Birth weight address t\_id

0 s001 V Alberto Franco 15/05/2002 35 street1 t1

1 s002 V Gino Mcneill 17/05/2002 32 street2 t2

2 s003 VI Ryan Parkes 16/02/1999 33 street3 t3

3 s001 VI Eesha Hinton 25/09/1998 30 street1 t4

4 s002 V Gino Mcneill 11/05/2002 31 street2 t5

5 s004 VI David Parkes 15/09/1997 32 street4 t6

school\_code as new Index:

school\_code class name date\_Of\_Birth weight address t\_id

s001 V Alberto Franco 15/05/2002 35 street1 t1

s002 V Gino Mcneill 17/05/2002 32 street2 t2

s003 VI Ryan Parkes 16/02/1999 33 street3 t3

s001 VI Eesha Hinton 25/09/1998 30 street1 t4

s002 V Gino Mcneill 11/05/2002 31 street2 t5

s004 VI David Parkes 15/09/1997 32 street4 t6

t\_id as new Index:

t\_id school\_code class name date\_Of\_Birth weight address

t1 s001 V Alberto Franco 15/05/2002 35 street1

t2 s002 V Gino Mcneill 17/05/2002 32 street2

t3 s003 VI Ryan Parkes 16/02/1999 33 street3

t4 s001 VI Eesha Hinton 25/09/1998 30 street1

t5 s002 V Gino Mcneill 11/05/2002 31 street2

t6 s004 VI David Parkes 15/09/1997 32 street4

**2. Write a Pandas program to create an index labels by using 64-bit integers, using floating-point numbers in a given dataframe.**

**Python Code :**

import pandas as pd

print("Create an Int64Index:")

df\_i64 = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']},

index=[1, 2, 3, 4, 5, 6])

print(df\_i64)

print("\nView the Index:")

print(df\_i64.index)

print("\nFloating-point labels using Float64Index:")

df\_f64 = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth ': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']},

index=[.1, .2, .3, .4, .5, .6])

print(df\_f64)

print("\nView the Index:")

print(df\_f64.index)

**Output:**

Create an Int64Index:

school\_code class name date\_Of\_Birth weight address

1 s001 V Alberto Franco 15/05/2002 35 street1

2 s002 V Gino Mcneill 17/05/2002 32 street2

3 s003 VI Ryan Parkes 16/02/1999 33 street3

4 s001 VI Eesha Hinton 25/09/1998 30 street1

5 s002 V Gino Mcneill 11/05/2002 31 street2

6 s004 VI David Parkes 15/09/1997 32 street4

View the Index:

Int64Index([1, 2, 3, 4, 5, 6], dtype='int64')

Floating-point labels using Float64Index:

school\_code class name date\_Of\_Birth weight address

0.1 s001 V Alberto Franco 15/05/2002 35 street1

0.2 s002 V Gino Mcneill 17/05/2002 32 street2

0.3 s003 VI Ryan Parkes 16/02/1999 33 street3

0.4 s001 VI Eesha Hinton 25/09/1998 30 street1

0.5 s002 V Gino Mcneill 11/05/2002 31 street2

0.6 s004 VI David Parkes 15/09/1997 32 street4

View the Index:

Float64Index([0.1, 0.2, 0.3, 0.4, 0.5, 0.6], dtype='float64')

**14.Pandas Strings and Regular Expressions:**

**1. Write a Pandas program to convert all the string values to upper, lower cases in a given pandas series. Also find the length of the string values.**

**Python Code :**

import pandas as pd

import numpy as np

s = pd.Series(['X', 'Y', 'Z', 'Aaba', 'Baca', np.nan, 'CABA', None, 'bird', 'horse', 'dog'])

print("Original series:")

print(s)

print("\nConvert all string values of the said Series to upper case:")

print(s.str.upper())

print("\nConvert all string values of the said Series to lower case:")

print(s.str.lower())

print("\nLength of the string values of the said Series:")

print(s.str.len())

**Output:**

Original series:

0 X

1 Y

2 Z

3 Aaba

4 Baca

5 NaN

6 CABA

7 None

8 bird

9 horse

10 dog

dtype: object

Convert all string values of the said Series to upper case:

0 X

1 Y

2 Z

3 AABA

4 BACA

5 NaN

6 CABA

7 None

8 BIRD

9 HORSE

10 DOG

dtype: object

Convert all string values of the said Series to lower case:

0 x

1 y

2 z

3 aaba

4 baca

5 NaN

6 caba

7 None

8 bird

9 horse

10 dog

dtype: object

Length of the string values of the said Series:

0 1.0

1 1.0

2 1.0

3 4.0

4 4.0

5 NaN

6 4.0

7 NaN

8 4.0

9 5.0

10 3.0

dtype: float64

**2. Write a Pandas program to remove whitespaces, left sided whitespaces and right sided whitespaces of the string values of a given pandas series.**

**Python Code :**

import pandas as pd

color1 = pd.Index([' Green', 'Black ', ' Red ', 'White', ' Pink '])

print("Original series:")

print(color1)

print("\nRemove whitespace")

print(color1.str.strip())

print("\nRemove left sided whitespace")

print(color1.str.lstrip())

print("\nRemove Right sided whitespace")

print(color1.str.rstrip())

**Output:**

Original series:

Index([' Green', 'Black ', ' Red ', 'White', ' Pink '], dtype='object')

Remove whitespace

Index(['Green', 'Black', 'Red', 'White', 'Pink'], dtype='object')

Remove left sided whitespace

Index(['Green', 'Black ', 'Red ', 'White', 'Pink '], dtype='object')

Remove Right sided whitespace

Index([' Green', 'Black', ' Red', 'White', ' Pink'], dtype='object')

**3. Write a Pandas program to count of occurrence of a specified substring in a DataFrame column.**

**Python Code :**

import pandas as pd

df = pd.DataFrame({

'name\_code': ['c001','c002','c022', 'c2002', 'c2222'],

'date\_of\_birth ': ['12/05/2002','16/02/1999','25/09/1998','12/02/2022','15/09/1997'],

'age': [18.5, 21.2, 22.5, 22, 23]

})

print("Original DataFrame:")

print(df)

print("\nCount occurrence of 2 in date\_of\_birth column:")

df['count'] = list(map(lambda x: x.count("2"), df['name\_code']))

print(df)

**Output:**

Original DataFrame:

name\_code date\_of\_birth age

0 c001 12/05/2002 18.5

1 c002 16/02/1999 21.2

2 c022 25/09/1998 22.5

3 c2002 12/02/2022 22.0

4 c2222 15/09/1997 23.0

Count occurrence of 2 in date\_of\_birth column:

name\_code date\_of\_birth age count

0 c001 12/05/2002 18.5 0

1 c002 16/02/1999 21.2 1

2 c022 25/09/1998 22.5 2

3 c2002 12/02/2022 22.0 2

4 c2222 15/09/1997 23.0 4

**4. Write a Pandas program to swap the cases of a specified character column in a given DataFrame.**

**Python Code :**

import pandas as pd

df = pd.DataFrame({

'company\_code': ['Abcd','EFGF', 'zefsalf', 'sdfslew', 'zekfsdf'],

'date\_of\_sale': ['12/05/2002','16/02/1999','25/09/1998','12/02/2022','15/09/1997'],

'sale\_amount': [12348.5, 233331.2, 22.5, 2566552.0, 23.0]

})

print("Original DataFrame:")

print(df)

print("\nSwapp cases in comapny\_code:")

df['swapped\_company\_code'] = list(map(lambda x: x.swapcase(), df['company\_code']))

print(df)

**Output:**

Original DataFrame:

company\_code date\_of\_sale sale\_amount

0 Abcd 12/05/2002 12348.5

1 EFGF 16/02/1999 233331.2

2 zefsalf 25/09/1998 22.5

3 sdfslew 12/02/2022 2566552.0

4 zekfsdf 15/09/1997 23.0

Swapp cases in comapny\_code:

company\_code date\_of\_sale sale\_amount swapped\_company\_code

0 Abcd 12/05/2002 12348.5 aBCD

1 EFGF 16/02/1999 233331.2 efgf

2 zefsalf 25/09/1998 22.5 ZEFSALF

3 sdfslew 12/02/2022 2566552.0 SDFSLEW

4 zekfsdf 15/09/1997 23.0 ZEKFSDF

**15. Pandas Joining and merging DataFrame:**

**1. Write a Pandas program to join the two given dataframes along rows and assign all data.**

**Test Data:**

student\_data1:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

student\_data2:

student\_id name marks

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

**Python Code :**

import pandas as pd

student\_data1 = pd.DataFrame({ 'student\_id': ['S1', 'S2', 'S3', 'S4', 'S5'],

'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],

'marks': [200, 210, 190, 222, 199]})

student\_data2 = pd.DataFrame({ 'student\_id': ['S4', 'S5', 'S6', 'S7', 'S8'],

'name': ['Scarlette Fisher', 'Carla Williamson', 'Dante Morse', 'Kaiser William', 'Madeeha Preston'],

'marks': [201, 200, 198, 219, 201]})

print("Original DataFrames:")

print(student\_data1)

print("-------------------------------------")

print(student\_data2)

print("\nJoin the said two dataframes along rows:")

result\_data = pd.concat([student\_data1, student\_data2])

print(result\_data)

**Output:**

Original DataFrames:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

-------------------------------------

student\_id name marks

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

Join the said two dataframes along rows:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

**2. Write a Pandas program to append a list of dictioneries or series to a existing DataFrame and display the combined data**

**Test Data:**

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

**Dictionary:**

student\_id S6

name Scarlette Fisher

marks 205

**Python Code :**

import pandas as pd

student\_data1 = pd.DataFrame({

'student\_id': ['S1', 'S2', 'S3', 'S4', 'S5'],

'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],

'marks': [200, 210, 190, 222, 199]})

s6 = pd.Series(['S6', 'Scarlette Fisher', 205], index=['student\_id', 'name', 'marks'])

dicts = [{'student\_id': 'S6', 'name': 'Scarlette Fisher', 'marks': 203},

{'student\_id': 'S7', 'name': 'Bryce Jensen', 'marks': 207}]

print("Original DataFrames:")

print(student\_data1)

print("\nDictionary:")

print(s6)

combined\_data = student\_data1.append(dicts, ignore\_index=True, sort=False)

print("\nCombined Data:")

print(combined\_data)

**Output:**

Original DataFrames:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

Dictionary:

student\_id S6

name Scarlette Fisher

marks 205

dtype: object

Combined Data:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

5 S6 Scarlette Fisher 203

6 S7 Bryce Jensen 207

**3. Write a Pandas program to join the two dataframes with matching records from both sides where available.**

**Test Data:**

student\_data1:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

student\_data2:

student\_id name marks

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

**Python Code :**

import pandas as pd

student\_data1 = pd.DataFrame({ 'student\_id': ['S1', 'S2', 'S3', 'S4', 'S5'],

'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],

'marks': [200, 210, 190, 222, 199]})

student\_data2 = pd.DataFrame({ 'student\_id': ['S4', 'S5', 'S6', 'S7', 'S8'],

'name': ['Scarlette Fisher', 'Carla Williamson', 'Dante Morse', 'Kaiser William', 'Madeeha Preston'],

'marks': [201, 200, 198, 219, 201]})

print("Original DataFrames:")

print(student\_data1)

print(student\_data2)

merged\_data = pd.merge(student\_data1, student\_data2, on='student\_id', how='outer')

print("Merged data (outer join):")

print(merged\_data)

**Output:**

Original DataFrames:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

student\_id name marks

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

Merged data (outer join):

student\_id name\_x marks\_x name\_y marks\_y

0 S1 Danniella Fenton 200.0 NaN NaN

1 S2 Ryder Storey 210.0 NaN NaN

2 S3 Bryce Jensen 190.0 NaN NaN

3 S4 Ed Bernal 222.0 Scarlette Fisher 201.0

4 S5 Kwame Morin 199.0 Carla Williamson 200.0

5 S6 NaN NaN Dante Morse 198.0

6 S7 NaN NaN Kaiser William 219.0

7 S8 NaN NaN Madeeha Preston 201.0

**16. Pandas Time Series:**

**1. Write a Pandas program to create**

**a) Datetime object for Jan 15 2012.  
b) Specific date and time of 9:20 pm.  
c) Local date and time.  
d) A date without time.  
e) Current date.  
f) Time from a datetime.  
g) Current local time.**

**Python Code :**

import datetime

from datetime import datetime

print("Datetime object for Jan 11 2012:")

print(datetime(2012, 1, 11))

print("\nSpecific date and time of 9:20 pm")

print(datetime(2011, 1, 11, 21, 20))

print("\nLocal date and time:")

print(datetime.now())

print("\nA date without time: ")

print(datetime.date(datetime(2012, 5, 22)))

print("\nCurrent date:")

print(datetime.now().date())

print("\nTime from a datetime:")

print(datetime.time(datetime(2012, 12, 15, 18, 12)))

print("\nCurrent local time:")

print(datetime.now().time())

**Output:**

Datetime object for Jan 11 2012:

2012-01-11 00:00:00

Specific date and time of 9:20 pm

2011-01-11 21:20:00

Local date and time:

2022-12-13 09:06:22.414154

A date without time:

2012-05-22

Current date:

2022-12-13

Time from a datetime:

18:12:00

Current local time:

09:06:22.414276

**2. Write a Pandas program to create a date from a given year, month, day and another date from a given string formats.**

**Python Code :**

from datetime import datetime

date1 = datetime(year=2020, month=12, day=25)

print("Date from a given year, month, day:")

print(date1)

from dateutil import parser

date2 = parser.parse("1st of January, 2021")

print("\nDate from a given string formats:")

print(date2)

**Output:**

Date from a given year, month, day:

2020-12-25 00:00:00

Date from a given string formats:

2021-01-01 00:00:00

**3. Write a Pandas program to create a time-series with two index labels and random values. Also print the type of the index.**

**Python Code :**

import pandas as pd

import numpy as np

import datetime

from datetime import datetime, date

dates = [datetime(2011, 9, 1), datetime(2011, 9, 2)]

print("Time-series with two index labels:")

time\_series = pd.Series(np.random.randn(2), dates)

print(time\_series)

print("\nType of the index:")

print(type(time\_series.index))

**Output:**

Time-series with two index labels:

2011-09-01 -1.028482

2011-09-02 -1.237705

dtype: float64

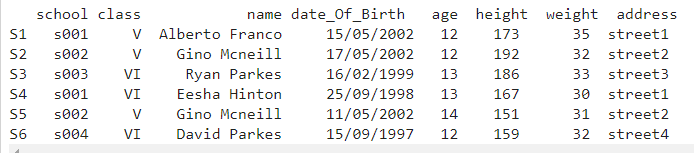
Type of the index:

<class 'pandas.core.indexes.datetimes.DatetimeIndex'>

**17. Pandas Grouping Aggregate:**

**1. Write a Pandas program to split the following dataframe into groups based on school code. Also check the type of GroupBy object.**

**Test Data:**



**Python Code :**

import pandas as pd

pd.set\_option('display.max\_rows', None)

#pd.set\_option('display.max\_columns', None)

student\_data = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth ': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'age': [12, 12, 13, 13, 14, 12],

'height': [173, 192, 186, 167, 151, 159],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']},

index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

print("Original DataFrame:")

print(student\_data)

print('\nSplit the said data on school\_code wise:')

result = student\_data.groupby(['school\_code'])

for name,group in result:

print("\nGroup:")

print(name)

print(group)

print("\nType of the object:")

print(type(result))

Output:

Original DataFrame:

school\_code class name ... height weight address

S1 s001 V Alberto Franco ... 173 35 street1

S2 s002 V Gino Mcneill ... 192 32 street2

S3 s003 VI Ryan Parkes ... 186 33 street3

S4 s001 VI Eesha Hinton ... 167 30 street1

S5 s002 V Gino Mcneill ... 151 31 street2

S6 s004 VI David Parkes ... 159 32 street4

[6 rows x 8 columns]

Split the said data on school\_code wise:

Group:

s001

school\_code class name ... height weight address

S1 s001 V Alberto Franco ... 173 35 street1

S4 s001 VI Eesha Hinton ... 167 30 street1

[2 rows x 8 columns]

Group:

s002

school\_code class name ... height weight address

S2 s002 V Gino Mcneill ... 192 32 street2

S5 s002 V Gino Mcneill ... 151 31 street2

[2 rows x 8 columns]

Group:

s003

school\_code class name ... height weight address

S3 s003 VI Ryan Parkes ... 186 33 street3

[1 rows x 8 columns]

Group:

s004

school\_code class name ... height weight address

S6 s004 VI David Parkes ... 159 32 street4

[1 rows x 8 columns]

Type of the object:

<class 'pandas.core.groupby.groupby.DataFrameGroupBy'>

**2. Write a Pandas program to split the following dataframe by school code and get mean, min, and max value of age for each school.**

**Python Code :**

import pandas as pd

pd.set\_option('display.max\_rows', None)

#pd.set\_option('display.max\_columns', None)

student\_data = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth ': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'age': [12, 12, 13, 13, 14, 12],

'height': [173, 192, 186, 167, 151, 159],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']},

index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

print("Original DataFrame:")

print(student\_data)

print('\nMean, min, and max value of age for each value of the school:')

grouped\_single = student\_data.groupby('school\_code').agg({'age': ['mean', 'min', 'max']})

print(grouped\_single)

**Output:**

Original DataFrame:

school\_code class name ... height weight address

S1 s001 V Alberto Franco ... 173 35 street1

S2 s002 V Gino Mcneill ... 192 32 street2

S3 s003 VI Ryan Parkes ... 186 33 street3

S4 s001 VI Eesha Hinton ... 167 30 street1

S5 s002 V Gino Mcneill ... 151 31 street2

S6 s004 VI David Parkes ... 159 32 street4

[6 rows x 8 columns]

Mean, min, and max value of age for each value of the school:

age

mean min max

school\_code

s001 12.5 12 13

s002 13.0 12 14

s003 13.0 13 13

s004 12.0 12 12

**18. Pandas Styling:**

**1. Create a dataframe of ten rows, four columns with random values. Write a Pandas program to highlight the negative numbers red and positive numbers black.**

**Python Code :**

import pandas as pd

import numpy as np

np.random.seed(24)

df = pd.DataFrame({'A': np.linspace(1, 10, 10)})

df = pd.concat([df, pd.DataFrame(np.random.randn(10, 4), columns=list('BCDE'))],

axis=1)

print("Original array:")

print(df)

def color\_negative\_red(val):

color = 'red' if val < 0 else 'black'

return 'color: %s' % color

print("\nNegative numbers red and positive numbers black:")

df.style.applymap(color\_negative\_red)

**2. Create a dataframe of ten rows, four columns with random values. Write a Pandas program to highlight the maximum value in each column.**

**Python Code :**

import pandas as pd

import numpy as np

np.random.seed(24)

df = pd.DataFrame({'A': np.linspace(1, 10, 10)})

df = pd.concat([df, pd.DataFrame(np.random.randn(10, 4), columns=list('BCDE'))],

axis=1)

df.iloc[0, 2] = np.nan

df.iloc[3, 3] = np.nan

df.iloc[4, 1] = np.nan

df.iloc[9, 4] = np.nan

print("Original array:")

print(df)

def highlight\_max(s):

'''

highlight the maximum in a Series green.

'''

is\_max = s == s.max()

return ['background-color: green' if v else '' for v in is\_max]

print("\nHighlight the maximum value in each column:")

df.style.apply(highlight\_max,subset=pd.IndexSlice[:, ['B', 'C', 'D', 'E']])

**3. Create a dataframe of ten rows, four columns with random values. Write a Pandas program to highlight dataframe's specific columns.**

**Python Code :**

import pandas as pd

import numpy as np

np.random.seed(24)

df = pd.DataFrame({'A': np.linspace(1, 10, 10)})

df = pd.concat([df, pd.DataFrame(np.random.randn(10, 4), columns=list('BCDE'))], axis=1)

df.iloc[0, 2] = np.nan

df.iloc[3, 3] = np.nan

df.iloc[4, 1] = np.nan

df.iloc[9, 4] = np.nan

print("Original array:")

print(df)

def highlight\_cols(s):

color = 'grey'

return 'background-color: %s' % color

print("\nHighlight specific columns:")

df.style.applymap(highlight\_cols, subset=pd.IndexSlice[:, ['B', 'C']])

**19. Excel:**

**1. Write a Pandas program to import excel data (coalpublic2013.xlsx ) into a Pandas dataframe.**

**Python Code :**

import pandas as pd

import numpy as np

df = pd.read\_excel('E:\coalpublic2013.xlsx')

print(df.head)

**2. Write a Pandas program to find the sum, mean, max, min value of 'Production (short tons)' column of coalpublic2013.xlsx file.**

**Python Code :**

import pandas as pd

import numpy as np

df = pd.read\_excel('E:\coalpublic2013.xlsx')

print("Sum: ",df["Production"].sum())

print("Mean: ",df["Production"].mean())

print("Maximum: ",df["Production"].max())

print("Minimum: ",df["Production"].min())

**20. Plotting:**

**1. Write a Pandas program to create a line plot of the historical stock prices of Alphabet Inc. between two specific dates**

**Python Code :**

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv("alphabet\_stock\_data.csv")

start\_date = pd.to\_datetime('2020-4-1')

end\_date = pd.to\_datetime('2020-09-30')

df['Date'] = pd.to\_datetime(df['Date'])

new\_df = (df['Date']>= start\_date) & (df['Date']<= end\_date)

df1 = df.loc[new\_df]

df2 = df1.set\_index('Date')

plt.figure(figsize=(5,5))

plt.suptitle('Stock prices of Alphabet Inc.,\n01-04-2020 to 30-09-2020', \ fontsize=18, color='black')

plt.xlabel("Date",fontsize=16, color='black')

plt.ylabel("$ price", fontsize=16, color='black')

df2['Close'].plot(color='green');

plt.show()

**2. Write a Pandas program to create a horizontal stacked bar plot of opening, closing stock prices of Alphabet Inc. between two specific dates.**

**Python Code :**

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv("alphabet\_stock\_data.csv")

start\_date = pd.to\_datetime('2020-4-1')

end\_date = pd.to\_datetime('2020-4-30')

df['Date'] = pd.to\_datetime(df['Date'])

new\_df = (df['Date']>= start\_date) & (df['Date']<= end\_date)

df1 = df.loc[new\_df]

df2 = df1[['Date', 'Open', 'Close']]

df3 = df2.set\_index('Date')

plt.figure(figsize=(20,20))

df3.plot.barh(stacked=True)

plt.suptitle('Opening/Closing stock prices Alphabet Inc.,\n01-04-2020 to 30-04-2020', fontsize=12, color='black')

plt.show()

**3. Write a Pandas program to create a histograms plot of opening, closing, high, low stock prices of Alphabet Inc. between two specific dates.**

**Python Code :**

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv("alphabet\_stock\_data.csv")

start\_date = pd.to\_datetime('2020-4-1')

end\_date = pd.to\_datetime('2020-9-30')

df['Date'] = pd.to\_datetime(df['Date'])

new\_df = (df['Date']>= start\_date) & (df['Date']<= end\_date)

df1 = df.loc[new\_df]

df2 = df1[['Open','Close','High','Low']]

#df3 = df2.set\_index('Date')

plt.figure(figsize=(25,25))

df2.plot.hist(alpha=0.5)

plt.suptitle('Opening/Closing/High/Low stock prices of Alphabet Inc.,\n From 01-04-2020 to 30-09-2020', fontsize=12, color='blue')

plt.show()

**4. Write a Pandas program to create a stacked histograms plot of opening, closing, high, low stock prices of Alphabet Inc. between two specific dates with more bins.**

**Python Code :**

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv("alphabet\_stock\_data.csv")

start\_date = pd.to\_datetime('2020-4-1')

end\_date = pd.to\_datetime('2020-9-30')

df['Date'] = pd.to\_datetime(df['Date'])

new\_df = (df['Date']>= start\_date) & (df['Date']<= end\_date)

df1 = df.loc[new\_df]

df2 = df1[['Open','Close','High','Low']]

plt.figure(figsize=(25,25))

df2.plot.hist(stacked=True, bins=200)

plt.suptitle('Opening/Closing/High/Low stock prices of Alphabet Inc.,\n From 01-04-2020 to 30-09-2020', fontsize=12, color='black')

plt.show()

**21. Pandas SQL Query:**

**1. Write a Pandas program to display all the records of a student file.**

**Python Code :**

import pandas as pd

employees = pd.read\_csv(r"EMPLOYEES.csv")

print("All the records from employees file:")

print(employees)

**2. Write a Pandas program to select distinct department id from employee's file.**

**Python Code :**

import pandas as pd

employees = pd.read\_csv(r"EMPLOYEES.csv")

departments = pd.read\_csv(r"DEPARTMENTS.csv")

print("Distinct department\_id:")

print(employees.department\_id.unique())