Query Processing for Data Sciences

DSA0505

Practical

1. Write a Pandas program to select distinct department id from employees file.

parthibhan R

192224275

import pandas as pd

data = {

'DEPARTMENT\_ID': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 260, 270],

'DEPARTMENT\_NAME': ['Administration', 'Marketing', 'Purchasing', 'Human Resources', 'Shipping', 'IT', 'Public Relations', 'Sales', 'Executive', 'Finance', 'Accounting', 'Treasury', 'Corporate Tax', 'Control And Credit', 'Shareholder Services', 'Benefits', 'Manufacturing', 'Construction', 'Contracting', 'Operations', 'IT Support', 'NOC', 'IT Helpdesk', 'Government Sales', 'Retail Sales', 'Recruiting', 'Payroll'],

'MANAGER\_ID': [200, 201, 114, 203, 121, 103, 204, 145, 100, 108, 205, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

'LOCATION\_ID': [1700, 1800, 1700, 2400, 1500, 1400, 2700, 2500, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700]

}

df = pd.DataFrame(data)

distinct\_department\_ids = df['DEPARTMENT\_ID'].unique()

print("Distinct Department IDs:")

print(distinct\_department\_ids)

Sample Output:-

Distinct Department IDs:

[ 10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180

190 200 210 220 230 240 250 260 270]

1. Write a Pandas program to display the ID for those employees who did two or more jobs in the past.

Parthibhan R

192224275

import pandas as pd

data = {

'EMPLOYEE\_ID': [102, 101, 101, 201, 114, 122, 200, 176, 176, 200],

'START\_DATE': ['2001-01-13', '1997-09-21', '2001-10-28', '2004-02-17', '2006-03-24', '2007-01-01', '1995-09-17', '2006-03-24', '2007-01-01', '2002-07-01'],

'END\_DATE': ['2006-07-24', '2001-10-27', '2005-03-15', '2007-12-19', '2007-12-31', '2007-12-31', '2001-06-17', '2006-12-31', '2007-12-31', '2006-12-31'],

'JOB\_ID': ['IT\_PROG', 'AC\_ACCOUNT', 'AC\_MGR', 'MK\_REP', 'ST\_CLERK', 'ST\_CLERK', 'AD\_ASST', 'SA\_REP', 'SA\_MAN', 'AC\_ACCOUNT'],

'DEPARTMENT\_ID': [60, 110, 110, 20, 50, 50, 90, 80, 80, 90],

}

df = pd.DataFrame(data)

employee\_job\_counts = df.groupby('EMPLOYEE\_ID')['JOB\_ID'].nunique()

employees\_with\_multiple\_jobs = employee\_job\_counts[employee\_job\_counts >= 2].index

result\_df = df[df['EMPLOYEE\_ID'].isin(employees\_with\_multiple\_jobs)]

print("Employees who did two or more jobs in the past:")

print(result\_df[['EMPLOYEE\_ID', 'JOB\_ID']])

Sample Output:-

Employees who did two or more jobs in the past:

EMPLOYEE\_ID JOB\_ID

1 101 AC\_ACCOUNT

2 101 AC\_MGR

6 200 AD\_ASST

7 176 SA\_REP

8 176 SA\_MAN

1. 200 AC\_ACCOUNT

3.Write a Pandas program to display the details of jobs in descending sequence on job title.

Parthibhan R

192224275

import pandas as pd

data = {

'JOB\_ID': ['AD\_PRES', 'AD\_VP', 'AD\_ASST', 'FI\_MGR', 'FI\_ACCOUNT', 'AC\_MGR', 'AC\_ACCOUNT', 'SA\_MAN', 'SA\_REP', 'PU\_MAN', 'PU\_CLERK', 'ST\_MAN', 'ST\_CLERK', 'SH\_CLERK', 'IT\_PROG', 'MK\_MAN', 'MK\_REP', 'HR\_REP', 'PR\_REP'],

'JOB\_TITLE': ['President', 'Administration Vice President', 'Administration Assistant', 'Finance Manager', 'Accountant', 'Accounting Manager', 'Public Accountant', 'Sales Manager', 'Sales Representative', 'Purchasing Manager', 'Purchasing Clerk', 'Stock Manager', 'Stock Clerk', 'Shipping Clerk', 'Programmer', 'Marketing Manager', 'Marketing Representative', 'Human Resources Representative', 'Public Relations Representative'],

'MIN\_SALARY': [20080, 15000, 3000, 8200, 4200, 8200, 4200, 10000, 6000, 8000, 2500, 5500, 2008, 2500, 4000, 9000, 4000, 4000, 4500],

'MAX\_SALARY': [40000, 30000, 6000, 16000, 9000, 16000, 9000, 20080, 12008, 15000, 5500, 8500, 5000, 5500, 10000, 15000, 9000, 9000, 10500],

}

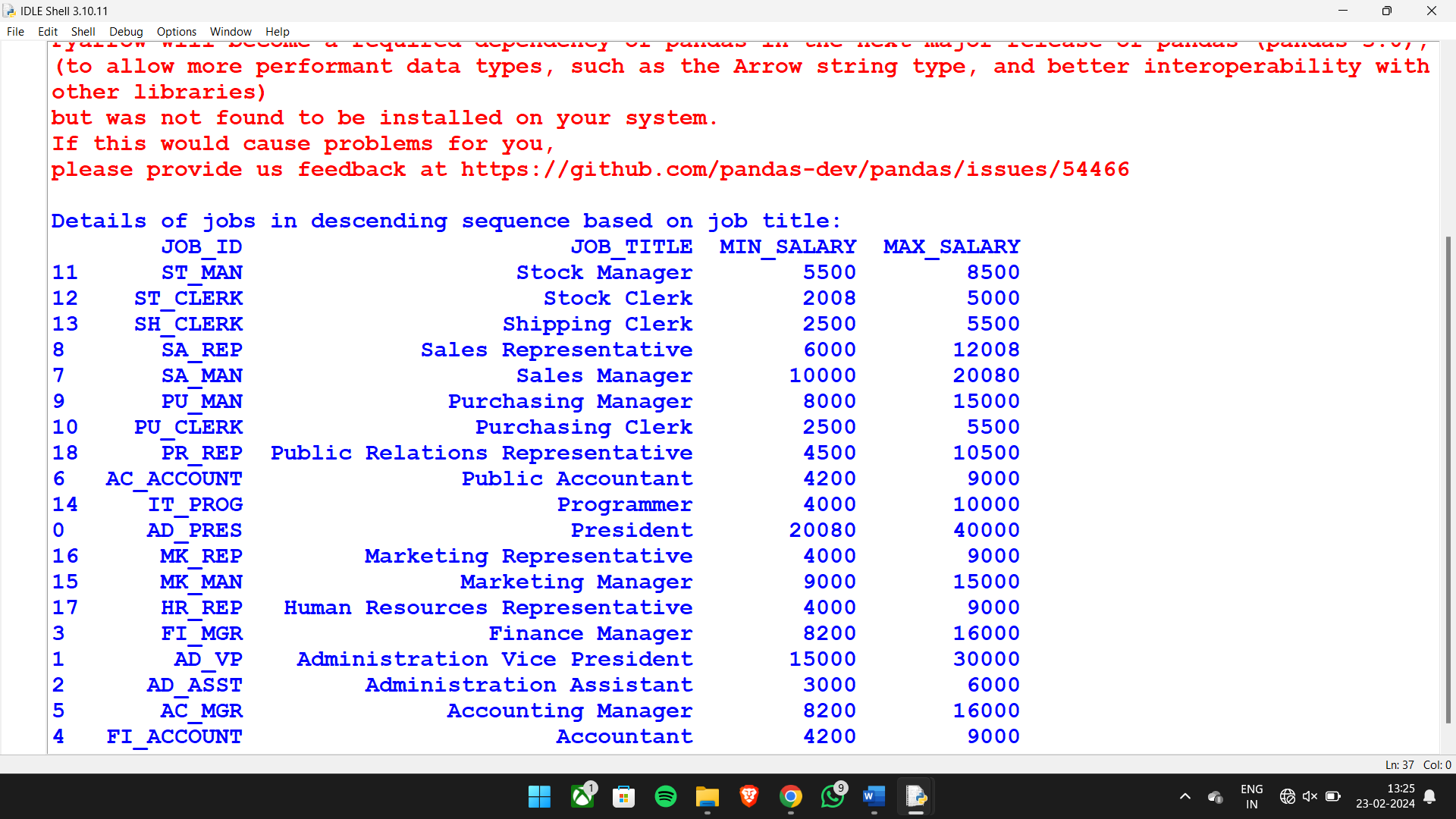
df = pd.DataFrame(data)

sorted\_df = df.sort\_values(by='JOB\_TITLE', ascending=False)

print("Details of jobs in descending sequence based on job title:")

print(sorted\_df)

Sample Output:-



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

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

start\_date = '2024-01-01'

end\_date = '2024-02-21'

num\_dates = 21

date\_range = pd.date\_range(start=start\_date, end=end\_date, periods=num\_dates)

prices = [100, 110, 105, 115, 120, 118, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195]

stock\_data = pd.DataFrame({'Date': date\_range, 'Price': prices})

stock\_data.plot(kind='line', x='Date', y='Price', figsize=(10, 6), color='blue')

plt.title('Historical Stock Prices of Alphabet Inc.')

plt.xlabel('Date')

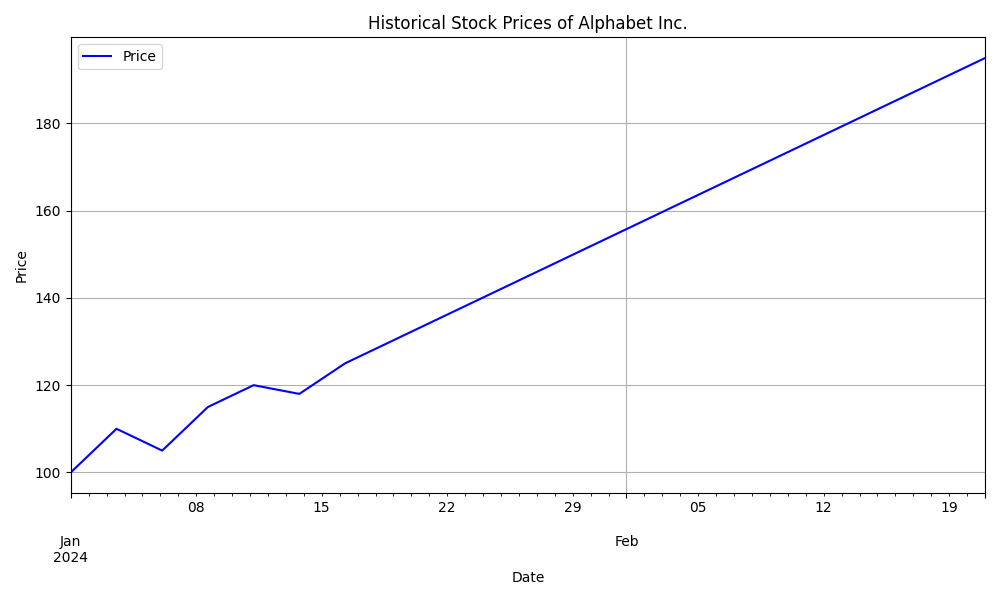
plt.ylabel('Price')

plt.grid(True)

plt.tight\_layout()

plt.show()

Sample Output:-



**5.** Write a Pandas program to create a bar plot of the trading volume of Alphabet Inc. stock between two specific dates.

#Parthibhan R

#192224275

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

start\_date = '2024-01-01'

end\_date = '2024-02-21'

num\_dates = 21

date\_range = pd.date\_range(start=start\_date, end=end\_date, periods=num\_dates)

volumes = [50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150]

volume\_data = pd.DataFrame({'Date': date\_range, 'Volume': volumes})

plt.figure(figsize=(10, 6))

plt.bar(volume\_data['Date'], volume\_data['Volume'], color='green')

plt.title('Trading Volume of Alphabet Inc.')

plt.xlabel('Date')

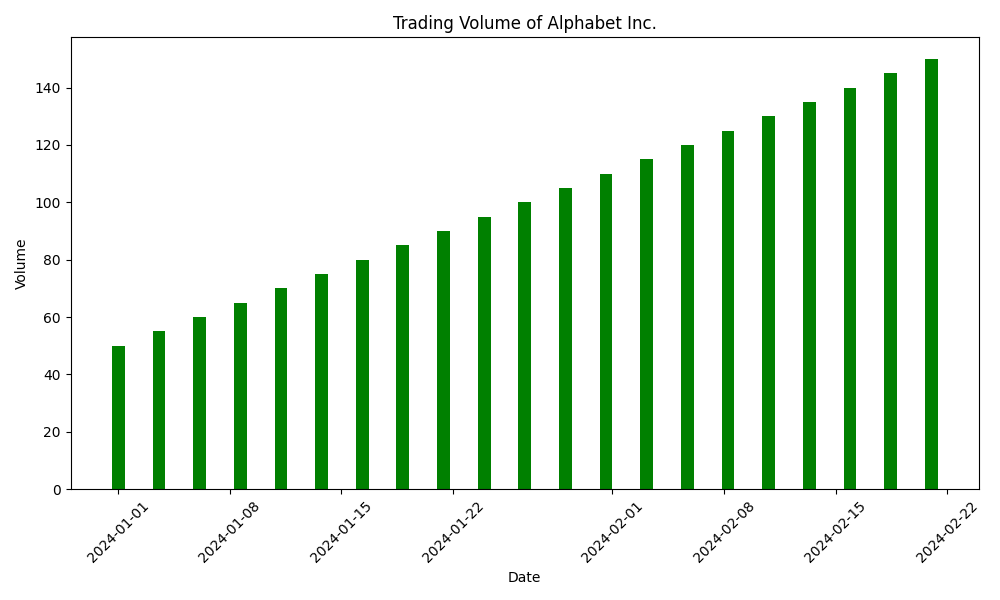
plt.ylabel('Volume')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

Sample Output:-



**6**.Write a Pandas program to create a scatter plot of the trading volume/stock prices of Alphabet Inc. stock between two specific dates.

#Parthibhan R

#192224275

import pandas as pd

import matplotlib.pyplot as plt

start\_date = '2024-01-01'

end\_date = '2024-02-21'

num\_dates = 21

date\_range = pd.date\_range(start=start\_date, end=end\_date, periods=num\_dates)

prices = [100, 110, 105, 115, 120, 118, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195]

volumes = [50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150]

stock\_data = pd.DataFrame({'Date': date\_range, 'Price': prices, 'Volume': volumes})

start\_date = '2024-01-10'

end\_date = '2024-02-10'

filtered\_data = stock\_data[(stock\_data['Date'] >= start\_date) & (stock\_data['Date'] <= end\_date)]

plt.figure(figsize=(10, 6))

plt.scatter(filtered\_data['Price'], filtered\_data['Volume'], color='blue')

plt.title('Trading Volume/Stock Prices of Alphabet Inc.')

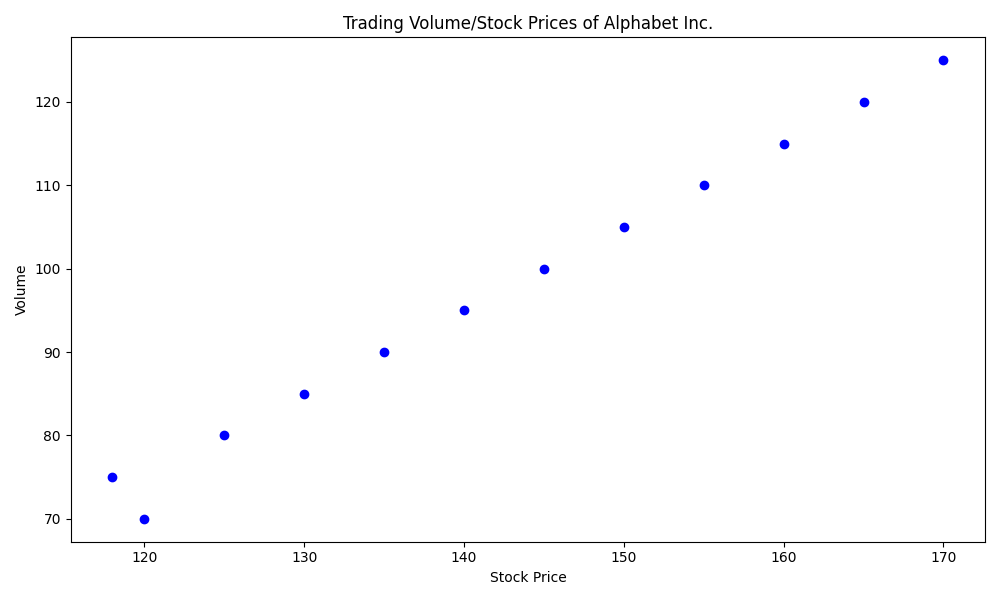
plt.xlabel('Stock Price')

plt.ylabel('Volume')

plt.tight\_layout()

plt.show()

**Sample Output:-**



**7**. Write a Pandas program to create a Pivot table and find the maximum and minimum sale value of the items.(refer sales\_data table)

#Parthibhan R

#192224275

import pandas as pd

data = {

'Item': ['A', 'B', 'A', 'B', 'A', 'B'],

'Date': ['2023-01-01', '2023-01-01', '2023-01-02', '2023-01-02', '2023-01-03', '2023-01-03'],

'Sale': [100, 150, 120, 130, 110, 160]

}

sales\_data = pd.DataFrame(data)

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

pivot\_table = pd.pivot\_table(sales\_data, values='Sale', index='Item', aggfunc=['max', 'min'])

pivot\_table.columns = ['Max Sale', 'Min Sale']

print("Pivot Table:")

print(pivot\_table)

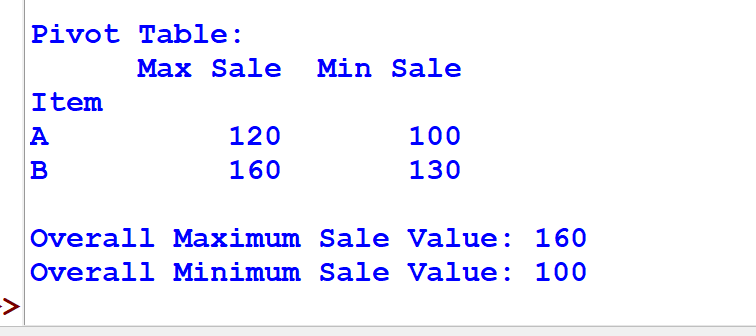
overall\_max\_sale = sales\_data['Sale'].max()

overall\_min\_sale = sales\_data['Sale'].min()

print("\nOverall Maximum Sale Value:", overall\_max\_sale)

print("Overall Minimum Sale Value:", overall\_min\_sale)

**Sample Output:-**



**8**. Write a Pandas program to create a Pivot table and find the item wise unit sold..(refer sales\_data table)

#Parthibhan R

#192224275

import pandas as pd

data = {

'Item': ['A', 'B', 'A', 'B', 'A', 'B'],

'Date': ['2023-01-01', '2023-01-01', '2023-01-02', '2023-01-02', '2023-01-03', '2023-01-03'],

'Unit Sold': [10, 15, 12, 13, 11, 16]

}

sales\_data = pd.DataFrame(data)

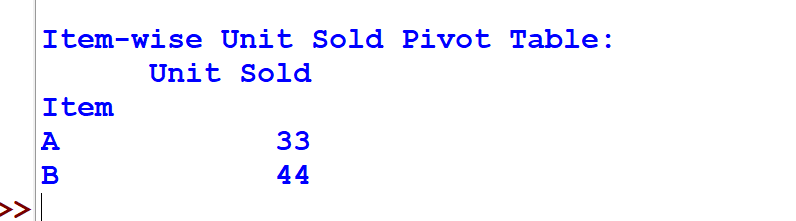
sales\_data['Date'] = pd.to\_datetime(sales\_data['Date'])

pivot\_table = pd.pivot\_table(sales\_data, values='Unit Sold', index='Item', aggfunc='sum')

print("Item-wise Unit Sold Pivot Table:")

print(pivot\_table)

**Sample Output:-**

****

**9.** Write a Pandas program to create a Pivot table and find the total sale amount region wise, manager wise, sales man wise..(refer sales\_data table)

**Sales\_data:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **OrderDate** | **Region** | **Manager** | **SalesMan** | **Item** | **Units** | **Unit\_price** | **Sale\_amt** |
| 1-6-18 | East | Martha | Alexander | Television | 95 | 1,198.00 | 1,13,810.00 |
| 1-23-18 | Central | Hermann | Shelli | Home Theater | 50 | 500.00 | 25,000.00 |
| 2-9-18 | Central | Hermann | Luis | Television | 36 | 1,198.00 | 43,128.00 |
| 2-26-18 | Central | Timothy | David | Cell Phone | 27 | 225.00 | 6,075.00 |
| 3-15-18 | West | Timothy | Stephen | Television | 56 | 1,198.00 | 67,088.00 |
| 4-1-18 | East | Martha | Alexander | Home Theater | 60 | 500.00 | 30,000.00 |
| 4-18-18 | Central | Martha | Steven | Television | 75 | 1,198.00 | 89,850.00 |
| 5-5-18 | Central | Hermann | Luis | Television | 90 | 1,198.00 | 1,07,820.00 |
| 5-22-18 | West | Douglas | Michael | Television | 32 | 1,198.00 | 38,336.00 |
| 6-8-18 | East | Martha | Alexander | Home Theater | 60 | 500.00 | 30,000.00 |
| 6-25-18 | Central | Hermann | Sigal | Television | 90 | 1,198.00 | 1,07,820.00 |
| 7-12-18 | East | Martha | Diana | Home Theater | 29 | 500.00 | 14,500.00 |
| 7-29-18 | East | Douglas | Karen | Home Theater | 81 | 500.00 | 40,500.00 |
| 8-15-18 | East | Martha | Alexander | Television | 35 | 1,198.00 | 41,930.00 |
| 9-1-18 | Central | Douglas | John | Desk | 2 | 125.00 | 250.00 |
| 9-18-18 | East | Martha | Alexander | Video Games | 16 | 58.50 | 936.00 |
| 10-5-18 | Central | Hermann | Sigal | Home Theater | 28 | 500.00 | 14,000.00 |
| 10-22-18 | East | Martha | Alexander | Cell Phone | 64 | 225.00 | 14,400.00 |

#Parthibhan R

#192224275

import pandas as pd

data = {

'OrderDate': ['1-6-18', '1-23-18', '2-9-18', '2-26-18', '3-15-18', '4-1-18', '4-18-18', '5-5-18', '5-22-18', '6-8-18', '6-25-18', '7-12-18', '7-29-18', '8-15-18', '9-1-18', '9-18-18', '10-5-18', '10-22-18'],

'Region': ['East', 'Central', 'Central', 'Central', 'West', 'East', 'Central', 'Central', 'West', 'East', 'Central', 'East', 'East', 'East', 'Central', 'East', 'Central', 'East'],

'Manager': ['Martha', 'Hermann', 'Hermann', 'Timothy', 'Timothy', 'Martha', 'Martha', 'Hermann', 'Douglas', 'Martha', 'Hermann', 'Martha', 'Douglas', 'Martha', 'Douglas', 'Martha', 'Hermann', 'Martha'],

'SalesMan': ['Alexander', 'Shelli', 'Luis', 'David', 'Stephen', 'Alexander', 'Steven', 'Luis', 'Michael', 'Alexander', 'Sigal', 'Diana', 'Karen', 'Alexander', 'John', 'Alexander', 'Sigal', 'Alexander'],

'Item': ['Television', 'Home Theater', 'Television', 'Cell Phone', 'Television', 'Home Theater', 'Television', 'Television', 'Television', 'Home Theater', 'Television', 'Home Theater', 'Home Theater', 'Television', 'Desk', 'Video Games', 'Home Theater', 'Cell Phone'],

'Units': [95, 50, 36, 27, 56, 60, 75, 90, 32, 60, 90, 29, 81, 35, 2, 16, 28, 64],

'Unit\_price': [1198.00, 500.00, 1198.00, 225.00, 1198.00, 500.00, 1198.00, 1198.00, 1198.00, 500.00, 1198.00, 500.00, 500.00, 1198.00, 125.00, 58.50, 500.00, 225.00],

'Sale\_amt': [113810.00, 25000.00, 43128.00, 6075.00, 67088.00, 30000.00, 89850.00, 107820.00, 38336.00, 30000.00, 107820.00, 14500.00, 40500.00, 41930.00, 250.00, 936.00, 14000.00, 14400.00]

}

sales\_data = pd.DataFrame(data)

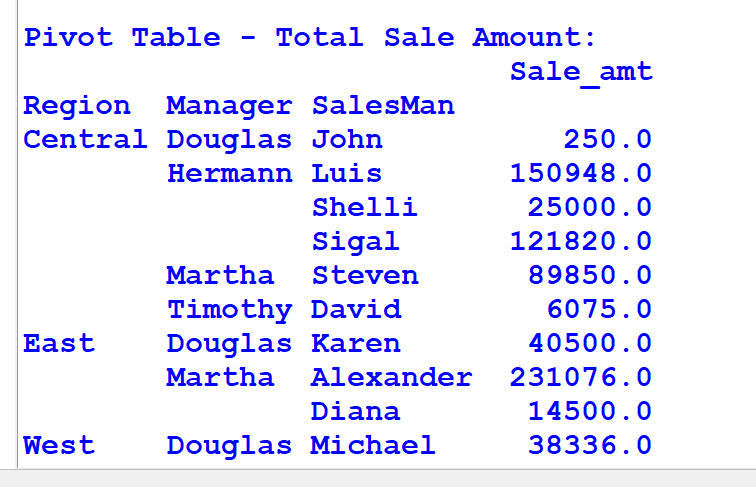
sales\_data['OrderDate'] = pd.to\_datetime(sales\_data['OrderDate'], format='%m-%d-%y')

pivot\_table = pd.pivot\_table(sales\_data, values='Sale\_amt', index=['Region', 'Manager', 'SalesMan'], aggfunc='sum')

print("Pivot Table - Total Sale Amount:")

print(pivot\_table)

**Sample Output:-**

****

10.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.

#Parthibhan R

#192224275

import pandas as pd

import numpy as np

np.random.seed(42)

data = np.random.randn(10, 4)

df = pd.DataFrame(data, columns=['A', 'B', 'C', 'D'])

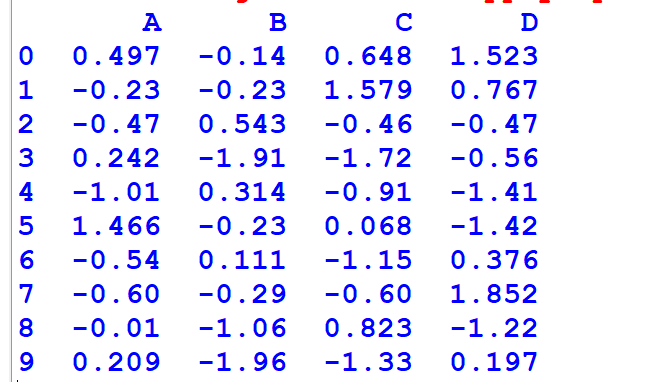
def format\_numbers(val):

return f'{val:.3f}' if val >= 0 else f'{val:.2f}'

formatted\_df = df.applymap(format\_numbers)

print(formatted\_df)

**Sample Output:-**



11.Create a dataframe of ten rows, four columns with random values. Convert some values to nan values. Write a Pandas program which will highlight the nan values.

#Parthibhan R

#192224275

import pandas as pd

import numpy as np

data = {

'A': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],

'B': [1.32921, np.nan, -1.07082, 0.564417, -1.6264, 0.961538, np.nan, 1.68658, -0.12982, -0.586538],

'C': [np.nan, -0.31628, -1.43871, 0.295722, 0.219565, np.nan, 0.104011, -1.32596, 0.631523, np.nan],

'D': [-0.99081, 1.88927, np.nan, np.nan, 1.05774, 0.562861, 1.2076, -2.08935, 1.39285, 0.519818],

'E': [0.678805, 1.03753, 1.6278, -0.385684, 0.850229, 0.165562, -0.00204021, np.nan, -0.063328, np.nan]

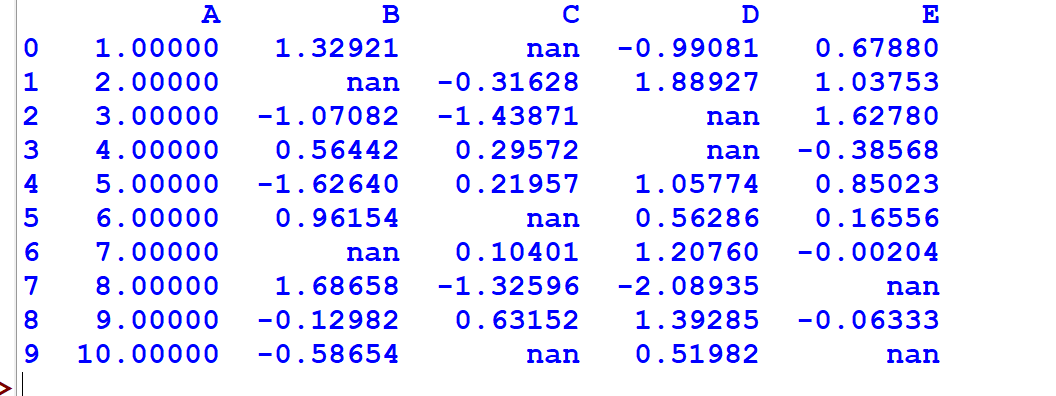
}

df = pd.DataFrame(data)

formatted\_df = df.applymap(lambda x: f'{x:.5f}' if not pd.isna(x) else 'nan')

print(formatted\_df)

**Sample Output:-**



12.Create a dataframe of ten rows, four columns with random values. Write a Pandas program to set dataframe background Color black and font color yellow.

import pandas as pd

import numpy as np

# Define the values for each dataset

values = [1.32111, 0.182, -1.8722, 17.988, np.nan]

# Define the number of datasets you want to create

num\_datasets = 1

# Create a list to store datasets

datasets = []

# Create multiple datasets

for \_ in range(num\_datasets):

# Create DataFrame with the specified values

df = pd.DataFrame([values] \* 5, columns=['A', 'B', 'C', 'D', 'E'])

# Append the DataFrame to the list of datasets

datasets.append(df)

# Display datasets

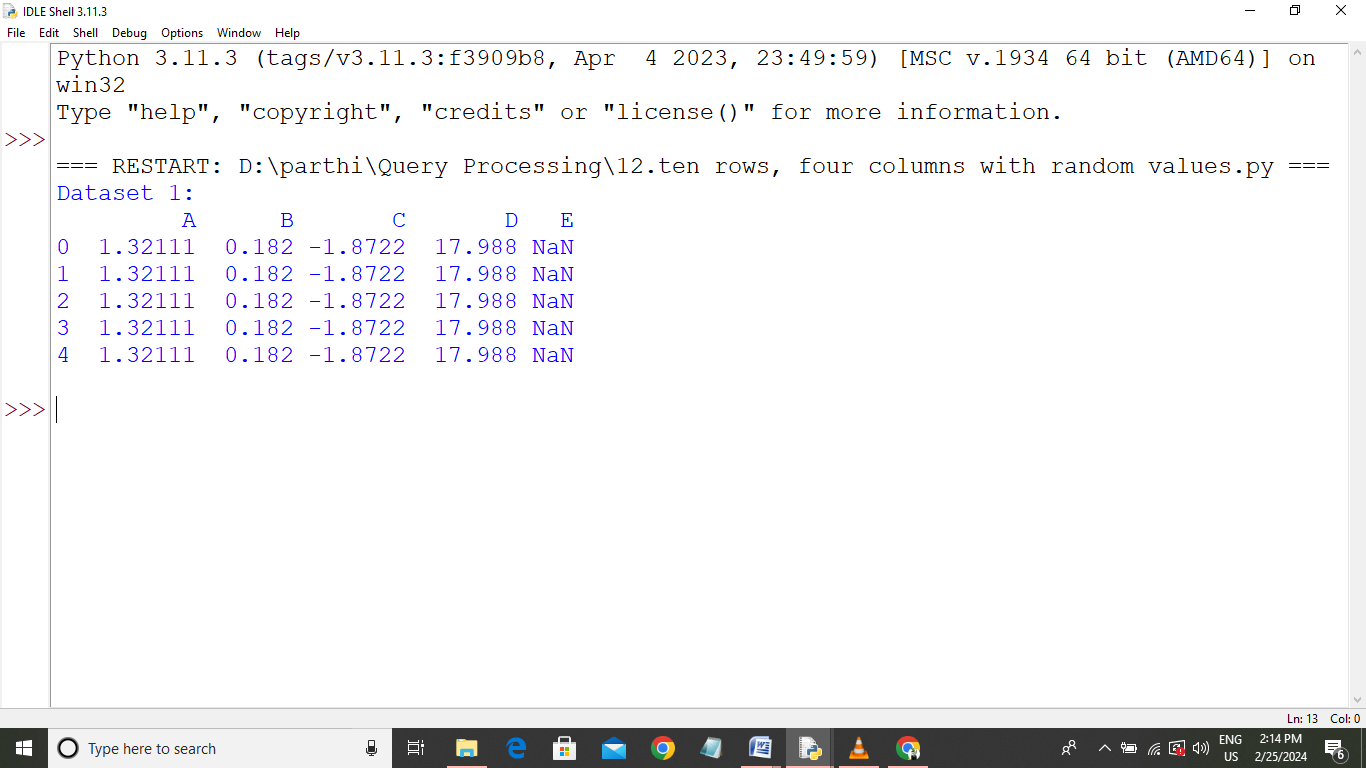
for i, df in enumerate(datasets):

print(f"Dataset {i+1}:")

print(df)

print()

**Sample Output:-**



13.Write a Pandas program to detect missing values of a given DataFrame. Display True or False.

import pandas as pd

import numpy as np

data = {'ord\_no': [70001.0, np.nan, 70002.0, 70004.0, np.nan, 70005.0, np.nan, 70010.0, 70003.0, 70012.0, np.nan, 70013.0],

'purch\_amt': [150.50, 270.65, 65.26, 110.50, 948.50, 2400.60, 5760.00, 1983.43, 2480.40, 250.45, 75.29, 3045.60],

'ord\_date': ['2012-10-05', '2012-09-10', np.nan, '2012-08-17', '2012-09-10', '2012-07-27', '2012-09-10', '2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', '2012-04-25'],

'customer\_id': [3002, 3001, 3001, 3003, 3002, 3001, 3001, 3004, 3003, 3002, 3001, 3001],

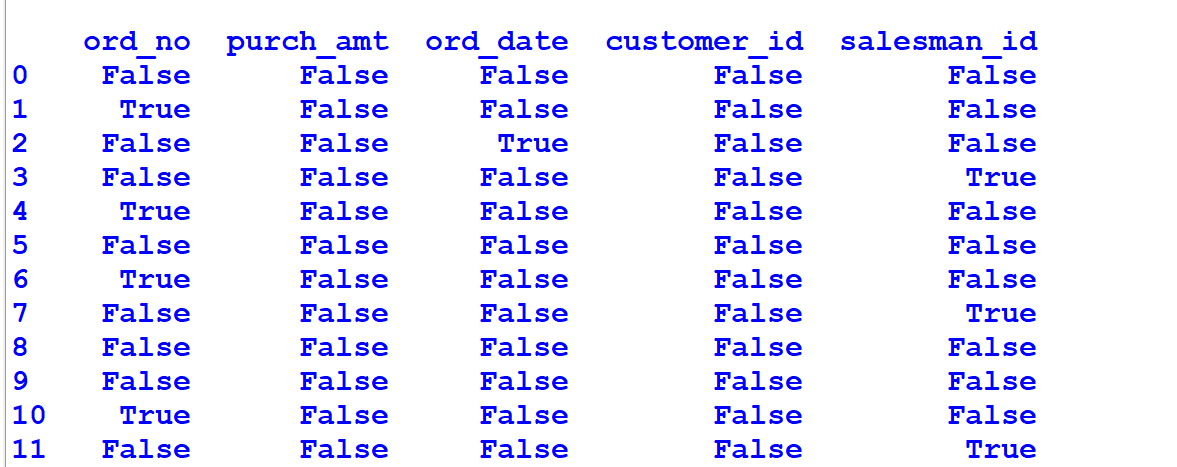
'salesman\_id': [5002.0, 5003.0, 5001.0, np.nan, 5002.0, 5001.0, 5001.0, np.nan, 5003.0, 5002.0, 5003.0, np.nan]}

df = pd.DataFrame(data)

missing\_values = df.isnull()

print(missing\_values)

**Sample Output:-**

****

14. Write a Pandas program to find and replace the missing values in a given DataFrame which do not have any valuable information.

import pandas as pd

import numpy as np

data = {'ord\_no': [70001.0, np.nan, 70002.0, 70004.0, np.nan, 70005.0, np.nan, 70010.0, 70003.0, 70012.0, np.nan, 70013.0],

'purch\_amt': [150.50, 270.65, 65.26, 110.50, 948.50, 2400.60, 5760.00, 1983.43, 2480.40, 250.45, 75.29, 3045.60],

'ord\_date': ['2012-10-05', '2012-09-10', np.nan, '2012-08-17', '2012-09-10', '2012-07-27', '2012-09-10', '2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', '2012-04-25'],

'customer\_id': [3002, 3001, 3001, 3003, 3002, 3001, 3001, 3004, 3003, 3002, 3001, 3001],

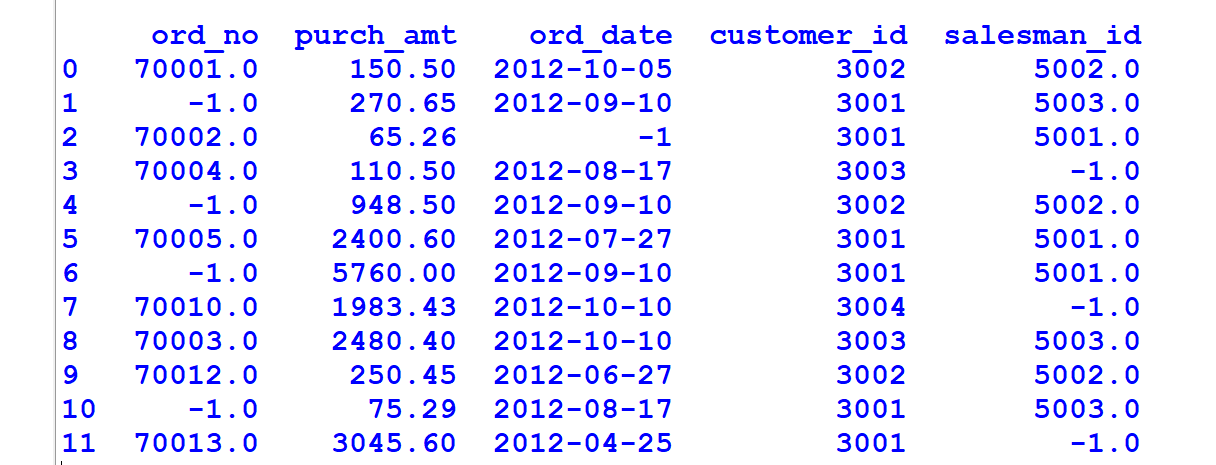
'salesman\_id': [5002.0, 5003.0, 5001.0, np.nan, 5002.0, 5001.0, 5001.0, np.nan, 5003.0, 5002.0, 5003.0, np.nan]}

df = pd.DataFrame(data)

df\_cleaned = df.fillna(-1)

print(df\_cleaned)

**Sample Output:-**



15.Write a Pandas program to keep the rows with at least 2 NaN values in a given DataFrame.

import pandas as pd

import numpy as np

data = {'ord\_no': [70001.0, np.nan, 70002.0, 70004.0, np.nan, 70005.0, np.nan, 70010.0, 70003.0, 70012.0, np.nan, 70013.0],

'purch\_amt': [150.50, 270.65, 65.26, 110.50, 948.50, 2400.60, 5760.00, 1983.43, 2480.40, 250.45, 75.29, 3045.60],

'ord\_date': ['2012-10-05', '2012-09-10', np.nan, '2012-08-17', '2012-09-10', '2012-07-27', '2012-09-10', '2012-10-10', '2012-10-10', '2012-06-27', '2012-08-17', '2012-04-25'],

'customer\_id': [3002, 3001, 3001, 3003, 3002, 3001, 3001, 3004, 3003, 3002, 3001, 3001],

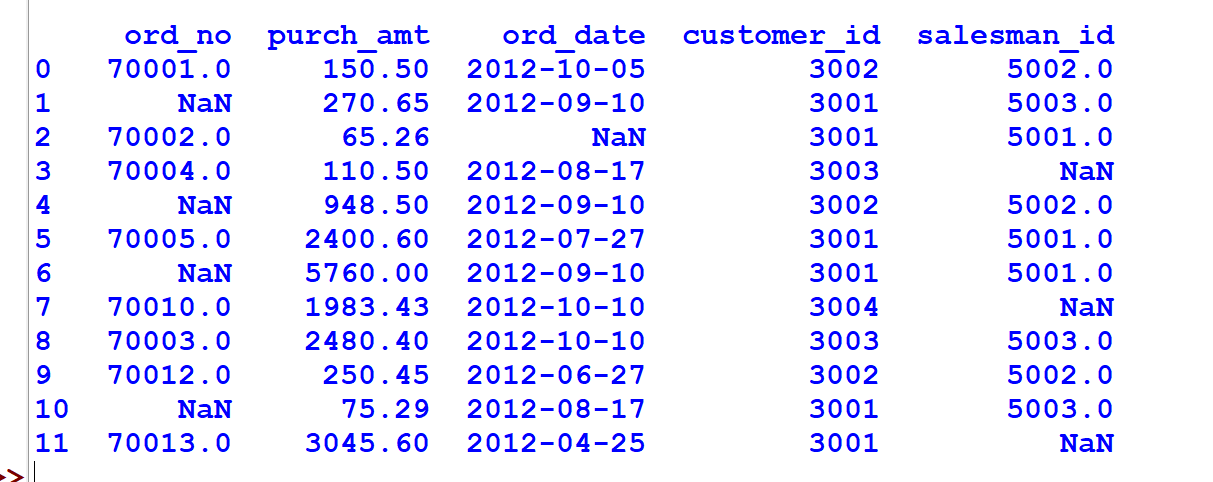
'salesman\_id': [5002.0, 5003.0, 5001.0, np.nan, 5002.0, 5001.0, 5001.0, np.nan, 5003.0, 5002.0, 5003.0, np.nan]}

df = pd.DataFrame(data)

df\_filtered = df.dropna(thresh=2)

print(df\_filtered)

**Sample Output:-**

****

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

import pandas as pd

from tabulate import tabulate

data = {

'Student\_ID': [1, 2, 3, 4, 5],

'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emily'],

'School\_Code': ['S001', 'S002', 'S001', 'S002', 'S003'],

'Grade': ['A', 'B', 'A', 'C', 'B']

}

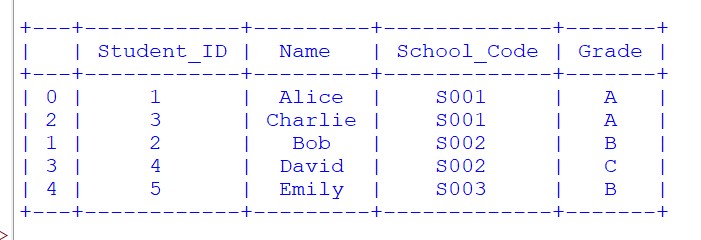
df = pd.DataFrame(data)

grouped = df.groupby('School\_Code')

all\_groups = pd.concat([group for \_, group in grouped])

print(tabulate(all\_groups, headers='keys', tablefmt='pretty'))

**Sample Output:-**

****

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

#Parthibhan R

#192224275

import pandas as pd

data = {

'Student\_ID': [1, 2, 3, 4, 5],

'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emily'],

'School\_Code': ['S001', 'S002', 'S001', 'S002', 'S003'],

'Age': [25, 30, 27, 28, 26]

}

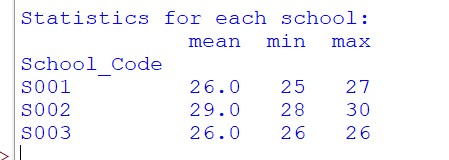
df = pd.DataFrame(data)

school\_stats = df.groupby('School\_Code')['Age'].agg(['mean', 'min', 'max'])

print("Statistics for each school:")

print(school\_stats)

**Sample Output:-**

**\**

18.Write a Pandas program to split the following given dataframe into groups based on school code and class.

import pandas as pd

data = {

'Student\_ID': [1, 2, 3, 4, 5, 6, 7, 8],

'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emily', 'Frank', 'Grace', 'Henry'],

'School\_Code': ['S001', 'S002', 'S001', 'S002', 'S003', 'S001', 'S002', 'S003'],

'Class': ['A', 'B', 'A', 'B', 'A', 'B', 'A', 'B']

}

df = pd.DataFrame(data)

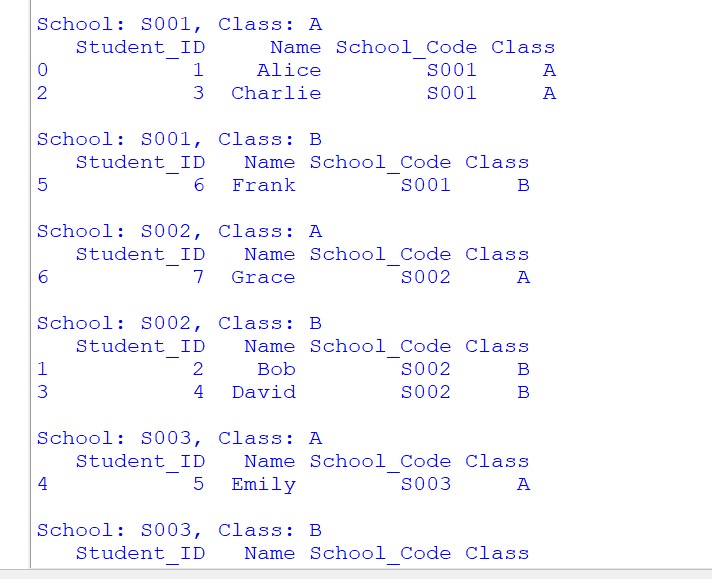
grouped = df.groupby(['School\_Code', 'Class'])

for (school, class\_), group in grouped:

print(f"\nSchool: {school}, Class: {class\_}")

print(group)

**Sample Output:-**

****

19.Write a Pandas program to display the dimensions or shape of the World alcohol consumption dataset. Also extract the column names from the dataset.

import pandas as pd

from io import StringIO

# CSV data

csv\_data = """

Year,WHO Region,Country,Type,Display Value

2015,Americas,United States,Beer,200

2015,Europe,France,Wine,300

2016,Africa,Nigeria,Spirits,150

2016,Americas,Canada,Beer,180

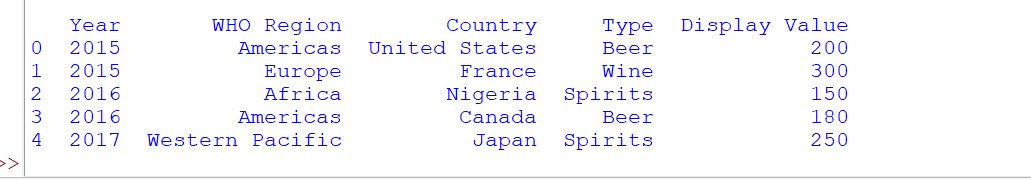
2017,Western Pacific,Japan,Spirits,250

"""

df = pd.read\_csv(StringIO(csv\_data))

print(df)

**Sample Output:-**

****

20.Write a Pandas program to find the index of a given substring of a DataFrame column.

#Parthibhan R

#192224275

import pandas as pd

data = {'Column1': ['apple', 'banana', 'orange', 'grape', 'kiwi']}

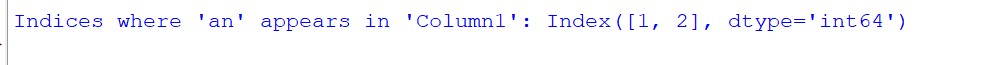
df = pd.DataFrame(data)

substring\_to\_find = 'an'

matching\_indices = df[df['Column1'].str.contains(substring\_to\_find)].index

print(f"Indices where '{substring\_to\_find}' appears in 'Column1': {matching\_indices}")

Sample Output:-



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

#Parthibhan R

#192224275

import pandas as pd

data = {'ID': [1, 2, 3, 4],

'Name': ['John', 'Alice', 'Bob', 'Eve']}

df = pd.DataFrame(data)

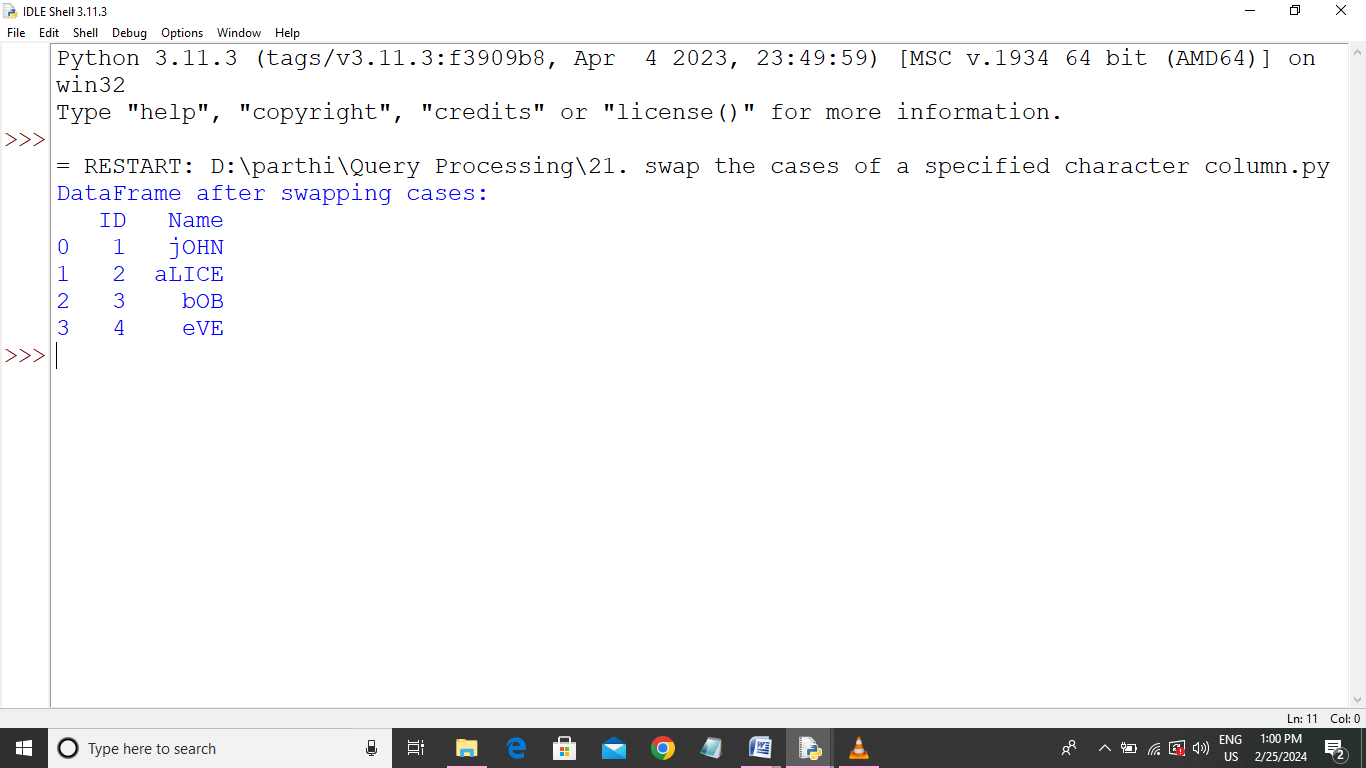
specified\_column = 'Name'

df[specified\_column] = df[specified\_column].str.swapcase()

print("DataFrame after swapping cases:")

print(df)

**Sample Output:-**

****

22.Write a Python program to draw a line with suitable label in the x axis, y axis and a title.

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

x\_values = [1, 2, 3, 4, 5]

y\_values = [2, 4, 6, 8, 10]

plt.plot(x\_values, y\_values)

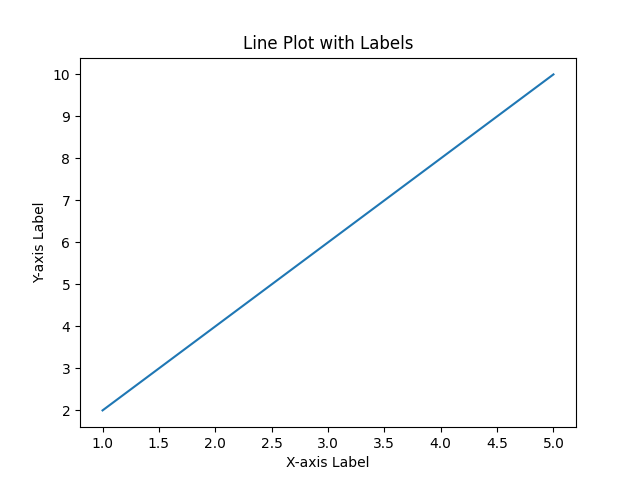
plt.xlabel('X-axis Label')

plt.ylabel('Y-axis Label')

plt.title('Line Plot with Labels')

plt.show()

**Sample Outtput:-**

****

23.Write a Python program to draw a line using given axis values taken from a text file, with suitable label in the x axis, y axis and a title.  
*Test Data:*

#Parthibhan R

#192224275

import pandas as pd

import matplotlib.pyplot as plt

data = {'X': [1, 2, 3],

'Y': [2, 4, 1]}

df = pd.DataFrame(data)

x\_values = df['X'].tolist()

y\_values = df['Y'].tolist()

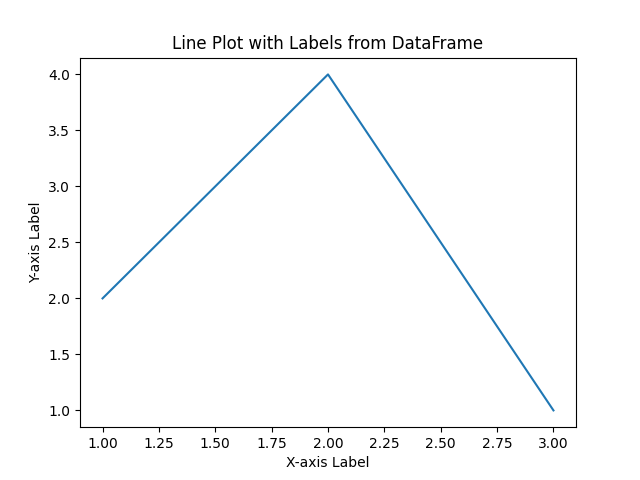
plt.plot(x\_values, y\_values)

plt.xlabel('X-axis Label')

plt.ylabel('Y-axis Label')

plt.title('Line Plot with Labels from DataFrame')

plt.show()

Sample Output:-

24.Write a Python program to draw line charts of the financial data of Alphabet Inc. between October 3, 2016 to October 7, 2016.  
Sample Financial data (fdata.csv):  
Date,Open,High,Low,Close  
10-03-16,774.25,776.065002,769.5,772.559998  
10-04-16,776.030029,778.710022,772.890015,776.429993  
10-05-16,779.309998,782.070007,775.650024,776.469971  
10-06-16,779,780.47998,775.539978,776.859985  
10-07-16,779.659973,779.659973,770.75,775.080017

#Parthibhan R

#192224275

import pandas as pd

import matplotlib.pyplot as plt

from io import StringIO

csv\_data = """Date,Open,High,Low,Close

10-03-16,774.25,776.065002,769.5,772.559998

10-04-16,776.030029,778.710022,772.890015,776.429993

10-05-16,779.309998,782.070007,775.650024,776.469971

10-06-16,779,780.47998,775.539978,776.859985

10-07-16,779.659973,779.659973,770.75,775.080017"""

df = pd.read\_csv(StringIO(csv\_data), parse\_dates=['Date'])

plt.plot(df['Date'], df['Open'], color='blue', label='Open', marker='o')

plt.plot(df['Date'], df['High'], color='green', label='High', marker='o')

plt.plot(df['Date'], df['Low'], color='red', label='Low', marker='o')

plt.plot(df['Date'], df['Close'], color='purple', label='Close', marker='o')

plt.xlabel('Date')

plt.ylabel('Price')

plt.title('Alphabet Inc. Financial Data (Oct 3, 2016 - Oct 7, 2016)')

plt.legend()

plt.show()

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv('fdata.csv', parse\_dates=['Date'])

plt.plot(df['Date'], df['Close'])

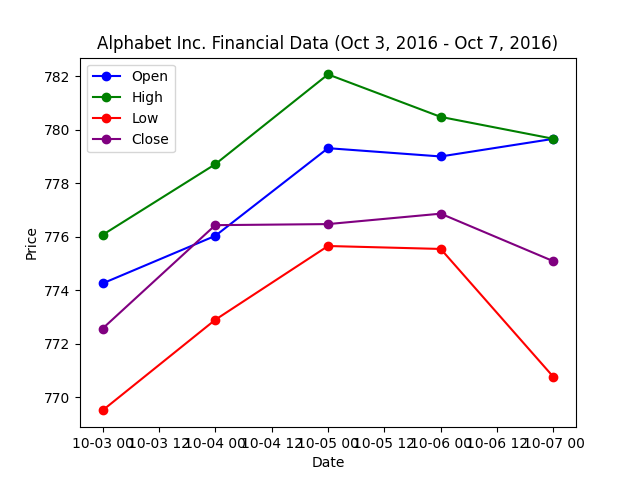
plt.xlabel('Date')

plt.ylabel('Closing Price')

plt.title('Alphabet Inc. Financial Data (Oct 3, 2016 - Oct 7, 2016)')

plt.show()'''

**Sample Output:-**

****

25.Write a Python program to plot two or more lines with legends, different widths and colors.

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

import numpy as np

x\_values = np.linspace(0, 10, 100)

y\_values\_1 = np.sin(x\_values)

y\_values\_2 = np.cos(x\_values)

plt.plot(x\_values, y\_values\_1, label='Line 1', color='blue', linewidth=3, linestyle='-', marker='D', markersize=8, markeredgecolor='blue', markerfacecolor='none')

plt.plot(x\_values, y\_values\_2, label='Line 2', color='green', linewidth=5, linestyle='-', marker='D', markersize=8, markeredgecolor='green', markerfacecolor='none')

plt.xlabel('X-axis')

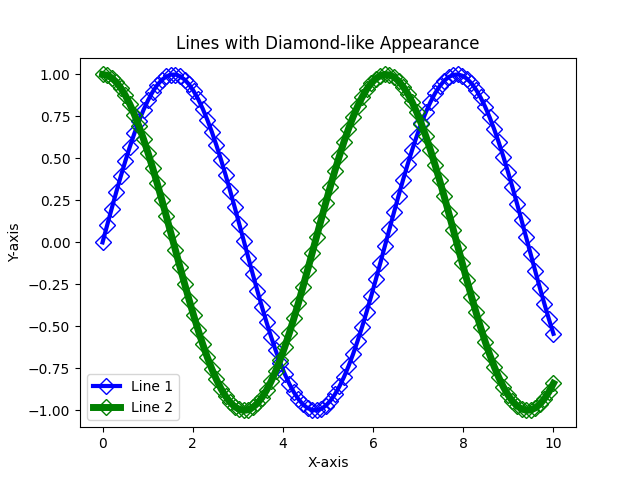
plt.ylabel('Y-axis')

plt.title('Lines with Diamond-like Appearance')

plt.legend()

plt.show()

**Sample Output:-**

****

26.Write a Python program to create multiple plots.

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(8, 8))

axes[0, 0].add\_patch(plt.Rectangle((0.1, 0.1), 0.8, 0.8, fill=None, edgecolor='blue'))

axes[0, 0].set\_title('Square 1')

axes[0, 1].add\_patch(plt.Rectangle((0.1, 0.1), 0.8, 0.8, fill=None, edgecolor='green'))

axes[0, 1].set\_title('Square 2')

axes[1, 0].add\_patch(plt.Rectangle((0.1, 0.1), 0.8, 0.8, fill=None, edgecolor='red'))

axes[1, 0].set\_title('Square 3')

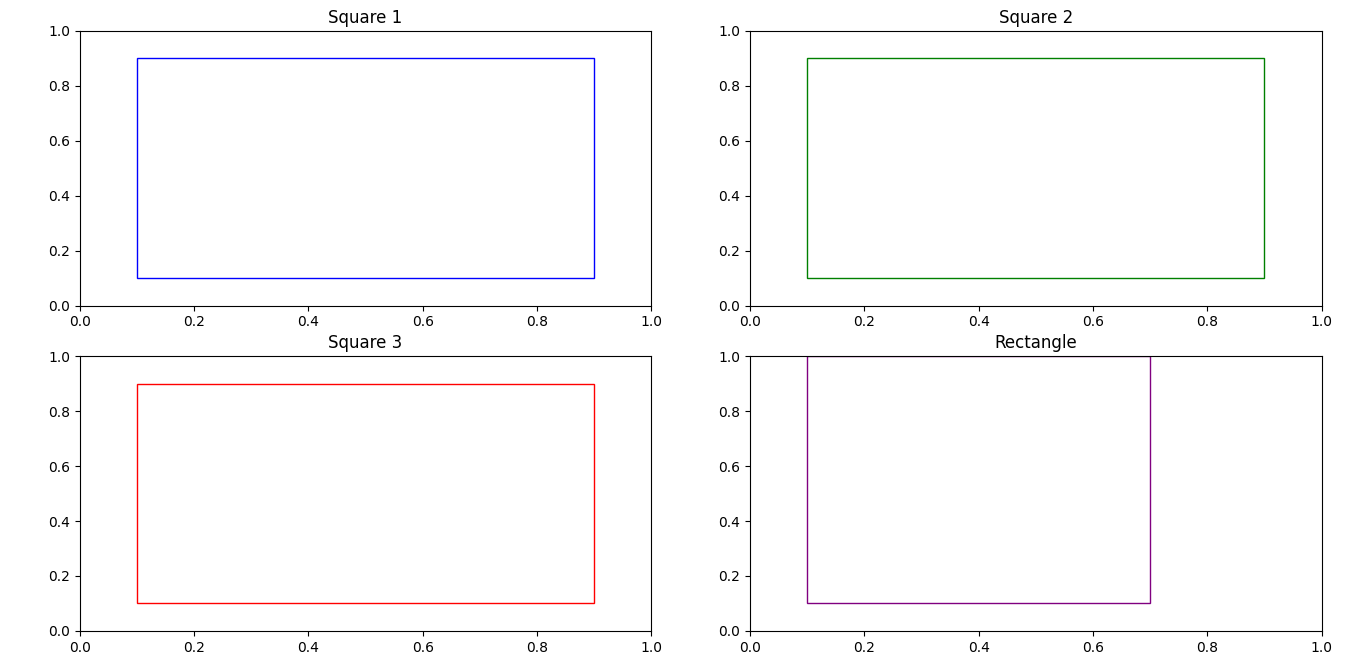
axes[1, 1].add\_patch(plt.Rectangle((0.1, 0.1), 0.6, 0.9, fill=None, edgecolor='purple'))

axes[1, 1].set\_title('Rectangle')

plt.tight\_layout()

plt.show()

**Sample Output:-**



27.Write a Python programming to display a bar chart of the popularity of programming Languages.  
Sample data:  
Programming languages: Java, Python, PHP, JavaScript, C#, C++  
Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']

popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

plt.bar(languages, popularity, color=['red', 'blue', 'green', 'yellow', 'purple', 'orange'])

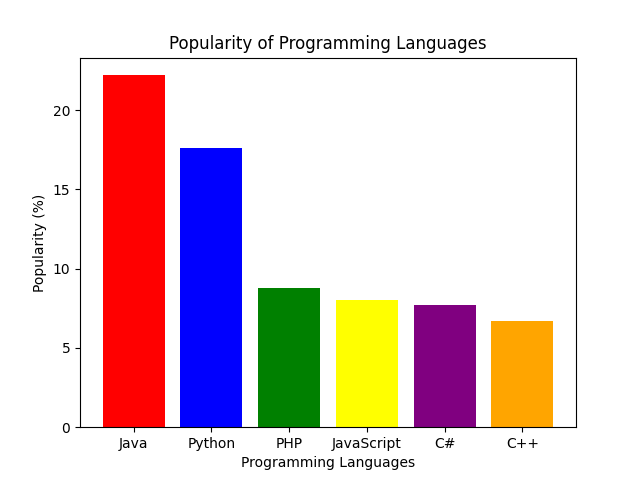
plt.xlabel('Programming Languages')

plt.ylabel('Popularity (%)')

plt.title('Popularity of Programming Languages')

plt.show()

**Sample output:-**

****

28.Write a Python programming to display a horizontal bar chart of the popularity of programming Languages.  
Sample data:  
Programming languages: Java, Python, PHP, JavaScript, C#, C++  
Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

import matplotlib.pyplot as plt

# Sample data

languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']

popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

# Creating a horizontal bar chart

plt.barh(languages, popularity, color='skyblue')

plt.xlabel('Popularity (%)')

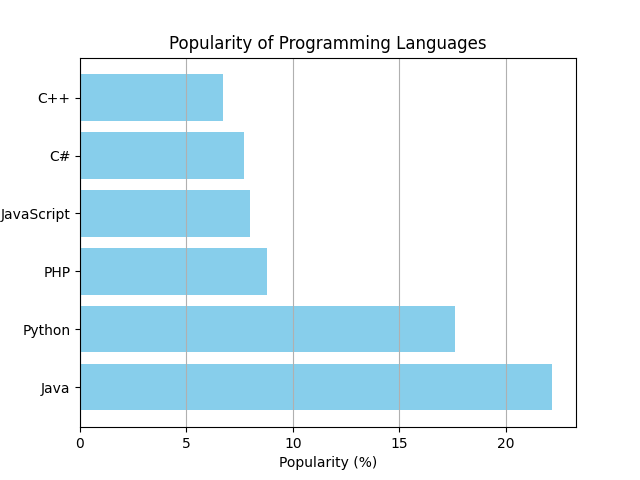
plt.title('Popularity of Programming Languages')

plt.grid(axis='x')

# Display the chart

plt.show()

**Sample Output:-**

****

29.Write a Python programming to display a bar chart of the popularity of programming Languages. Use different color for each bar.  
Sample data:  
Programming languages: Java, Python, PHP, JavaScript, C#, C++  
Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

import matplotlib.pyplot as plt

# Sample data

languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']

popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

colors = ['blue', 'green', 'red', 'yellow', 'purple', 'orange']

# Creating a bar chart with different colors

plt.bar(languages, popularity, color=colors)

plt.xlabel('Programming Languages')

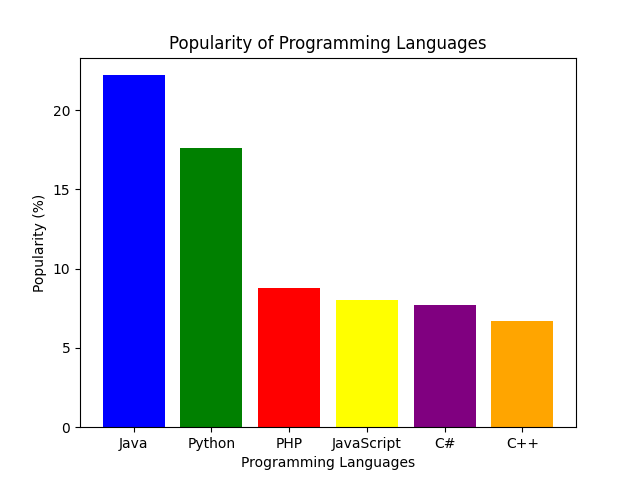
plt.ylabel('Popularity (%)')

plt.title('Popularity of Programming Languages')

# Display the chart

plt.show()

**Sample Output:-**

****

30.Write a Python program to create bar plot of scores by group and gender. Use multiple X values on the same chart for men and women.  
  
Sample Data:  
Means (men) = (22, 30, 35, 35, 26)  
Means (women) = (25, 32, 30, 35, 29)

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

import numpy as np

groups = ['Group 1', 'Group 2', 'Group 3', 'Group 4', 'Group 5']

means\_men = [22, 30, 35, 35, 26]

means\_women = [25, 32, 30, 35, 29]

x\_values\_men = np.arange(len(groups))

x\_values\_women = x\_values\_men + 0.4

plt.bar(x\_values\_men, means\_men, width=0.4, label='Men', color='blue')

plt.bar(x\_values\_women, means\_women, width=0.4, label='Women', color='pink')

plt.xlabel('Groups')

plt.ylabel('Scores')

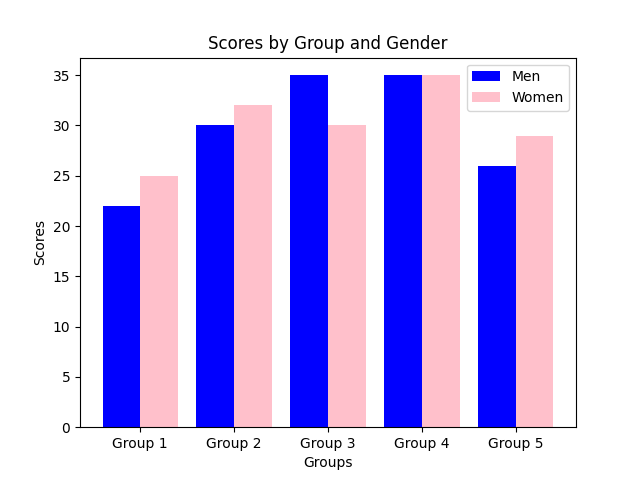
plt.title('Scores by Group and Gender')

plt.xticks(x\_values\_men + 0.2, groups)

plt.legend()

plt.show()

**Sample Output:-**

****

31.Write a Python program to create a stacked bar plot with error bars.  
Note: Use bottom to stack the women?s bars on top of the men?s bars.  
Sample Data:  
Means (men) = (22, 30, 35, 35, 26)  
Means (women) = (25, 32, 30, 35, 29)  
Men Standard deviation = (4, 3, 4, 1, 5)  
Women Standard deviation = (3, 5, 2, 3, 3)

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

import numpy as np

groups = ['Group 1', 'Group 2', 'Group 3', 'Group 4', 'Group 5']

means\_men = [22, 30, 35, 35, 26]

means\_women = [25, 32, 30, 35, 29]

std\_dev\_men = [4, 3, 4, 1, 5]

std\_dev\_women = [3, 5, 2, 3, 3]

x\_values = np.arange(len(groups))

plt.bar(x\_values, means\_men, yerr=std\_dev\_men, label='Men', color='blue', alpha=0.7)

plt.bar(x\_values, means\_women, yerr=std\_dev\_women, bottom=means\_men, label='Women', color='pink', alpha=0.7)

plt.xlabel('Groups')

plt.ylabel('Scores')

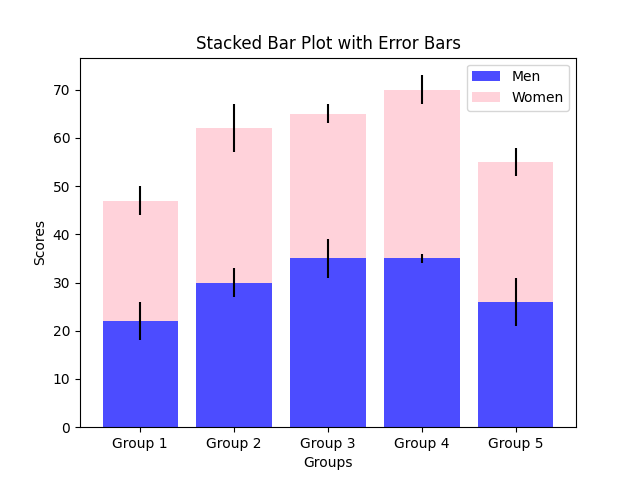
plt.title('Stacked Bar Plot with Error Bars')

plt.xticks(x\_values, groups)

plt.legend()

plt.show()

**Sample Output:-**



32.Write a Python program to draw a scatter graph taking a random distribution in X and Y and plotted against each other.

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

import numpy as np

np.random.seed(42)

num\_points = 50

x\_values = np.random.rand(num\_points)

y\_values = np.random.rand(num\_points)

plt.scatter(x\_values, y\_values, color='blue', marker='o', s=50, alpha=0.8)

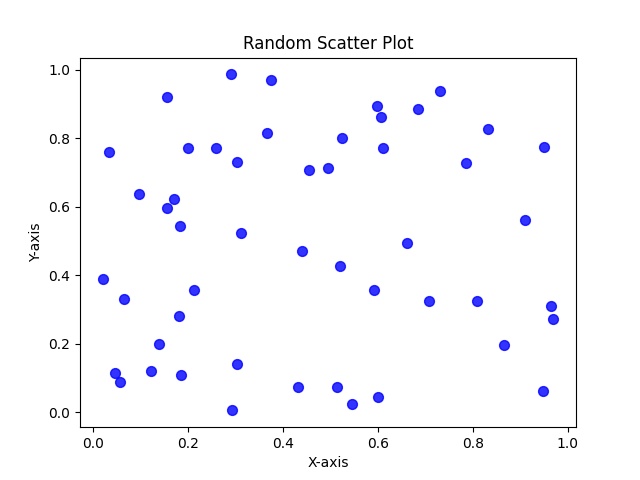
plt.xlabel('X-axis')

plt.ylabel('Y-axis')

plt.title('Random Scatter Plot')

plt.show()

**Sample Output**:-



33.Write a Python program to draw a scatter plot with empty circles taking a random distribution in X and Y and plotted against each other.

import matplotlib.pyplot as plt

import numpy as np

np.random.seed(42)

num\_points = 50

x\_values = np.random.rand(num\_points)

y\_values = np.random.rand(num\_points)

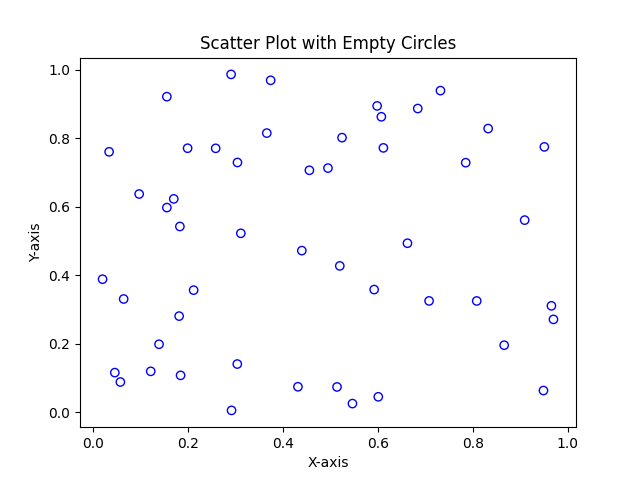
plt.scatter(x\_values, y\_values, facecolors='none', edgecolors='b', marker='o')

plt.xlabel('X-axis')

plt.ylabel('Y-axis')

plt.title('Scatter Plot with Empty Circles')

plt.show()

**Sample Output:-**

34.Write a Python program to draw a scatter plot using random distributions to generate balls of different sizes.

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

import numpy as np

np.random.seed(42)

num\_points = 50

x\_values = np.random.rand(num\_points)

y\_values = np.random.rand(num\_points)

ball\_sizes = np.random.randint(10, 100, num\_points)

ball\_colors = np.random.rand(num\_points, 3)

plt.scatter(x\_values, y\_values, c=ball\_colors, s=ball\_sizes, alpha=0.8)

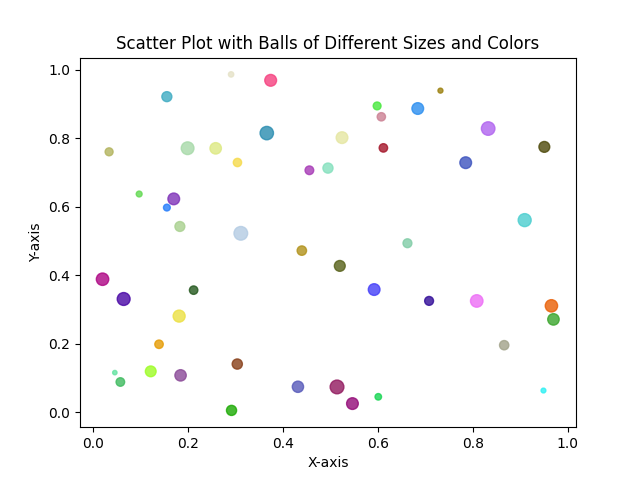
plt.xlabel('X-axis')

plt.ylabel('Y-axis')

plt.title('Scatter Plot with Balls of Different Sizes and Colors')

plt.show()

**Sample Output:-**



35.Write a Python program to draw a scatter plot comparing two subject marks of Mathematics and Science. Use marks of 10 students.  
Sample data:

Test Data:  
math\_marks = [88, 92, 80, 89, 100, 80, 60, 100, 80, 34]  
science\_marks = [35, 79, 79, 48, 100, 88, 32, 45, 20, 30]  
marks\_range = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

#Parthibhan R

#1922242755

import matplotlib.pyplot as plt

math\_marks = [88, 92, 80, 89, 100, 80, 60, 100, 80, 34]

science\_marks = [35, 79, 79, 48, 100, 88, 32, 45, 20, 30]

marks\_range = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

plt.scatter(marks\_range, math\_marks, color='blue', label='Mathematics', s=50, alpha=0.8)

plt.scatter(marks\_range, science\_marks, color='green', label='Science', s=50, alpha=0.8)

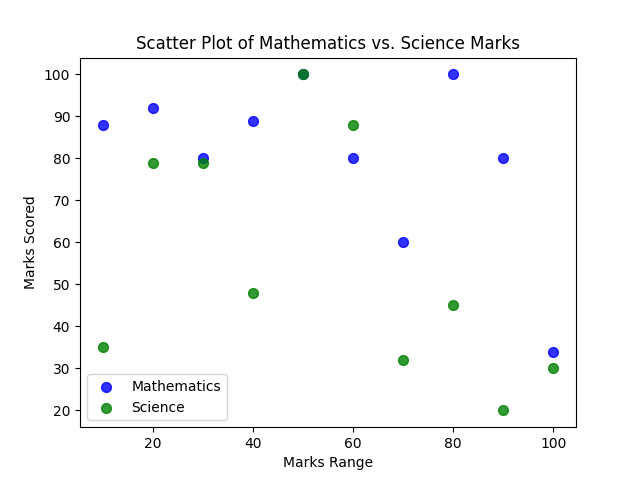
plt.xlabel('Marks Range')

plt.ylabel('Marks Scored')

plt.title('Scatter Plot of Mathematics vs. Science Marks')

plt.legend()

plt.show()

Sample Output:-

36.Write a Python program to draw a scatter plot for three different groups comparing weights and heights.

#Parthibhan R

#192224275

import matplotlib.pyplot as plt

import numpy as np

group1\_heights = np.random.normal(170, 10, 30)

group1\_weights = np.random.normal(65, 5, 30)

group2\_heights = np.random.normal(160, 8, 30)

group2\_weights = np.random.normal(55, 4, 30)

group3\_heights = np.random.normal(175, 12, 30)

group3\_weights = np.random.normal(75, 6, 30)

plt.scatter(group1\_weights, group1\_heights, color='Blue', label='Group 1', marker='\*', s=50, alpha=0.8)

plt.scatter(group2\_weights, group2\_heights, color='Blue', label='Group 2', marker='\*', s=50, alpha=0.8)

plt.scatter(group3\_weights, group3\_heights, color='Blue', label='Group 3', marker='\*', s=50, alpha=0.8)

plt.xlabel('Weights (kg)')

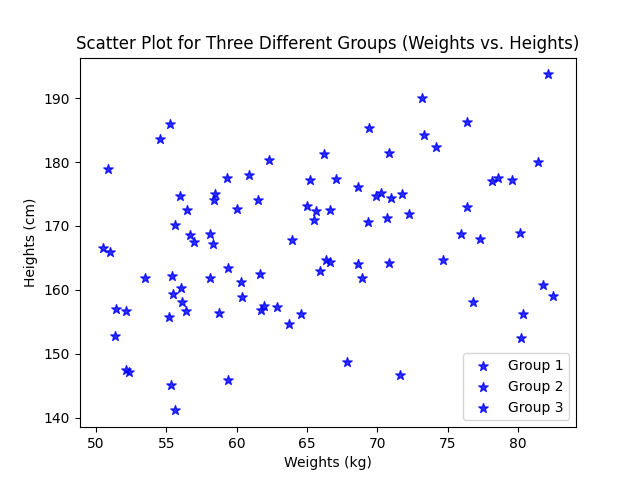
plt.ylabel('Heights (cm)')

plt.title('Scatter Plot for Three Different Groups (Weights vs. Heights)')

plt.legend()

plt.show()

Sample Output:-



37.Write a Pandas program to create a dataframe from a dictionary and display it.  
Sample data: {'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':[86,97,96,72,83]}

#Parthibhan R

#192224275

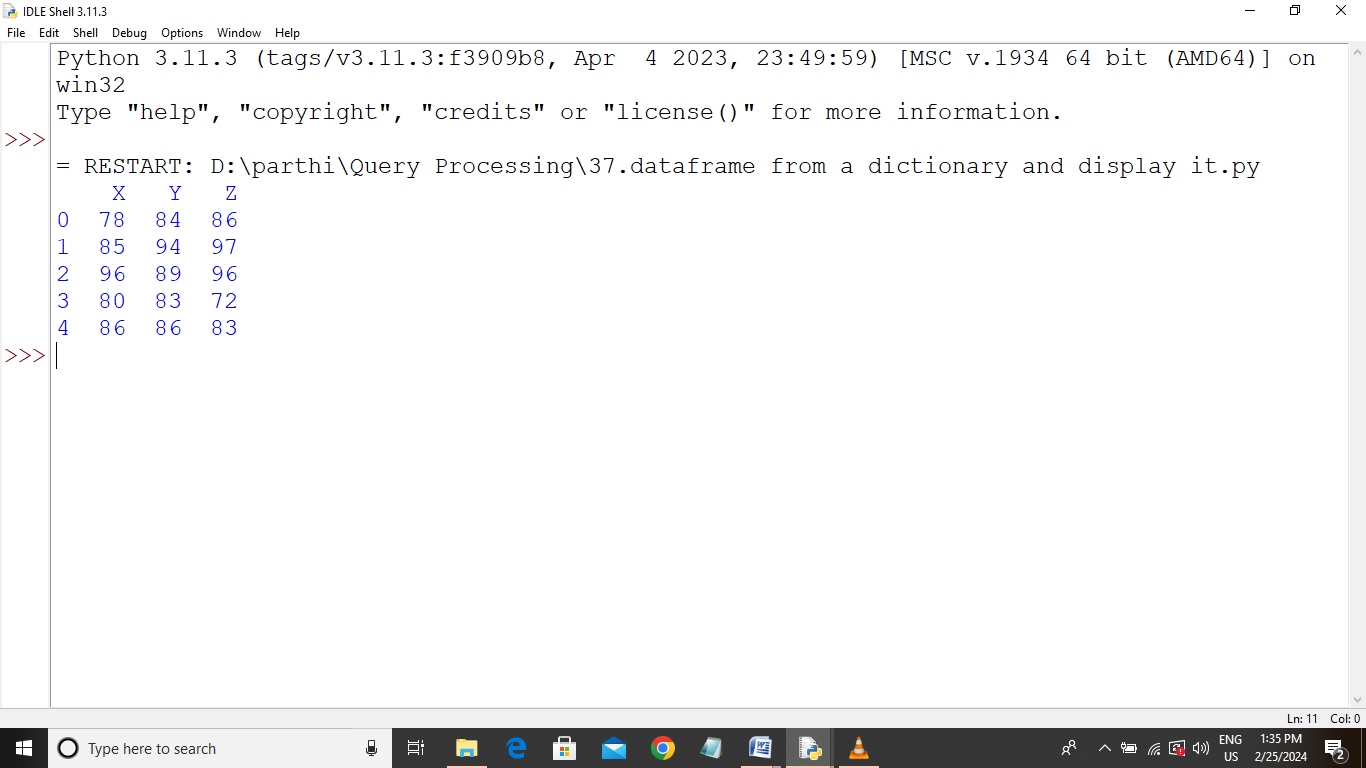
import pandas as pd

data = {'X': [78, 85, 96, 80, 86], 'Y': [84, 94, 89, 83, 86], 'Z': [86, 97, 96, 72, 83]}

df = pd.DataFrame(data)

print(df)

**Sample Output:-**

****

38.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']

#Parthibhan R

#192224275

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']

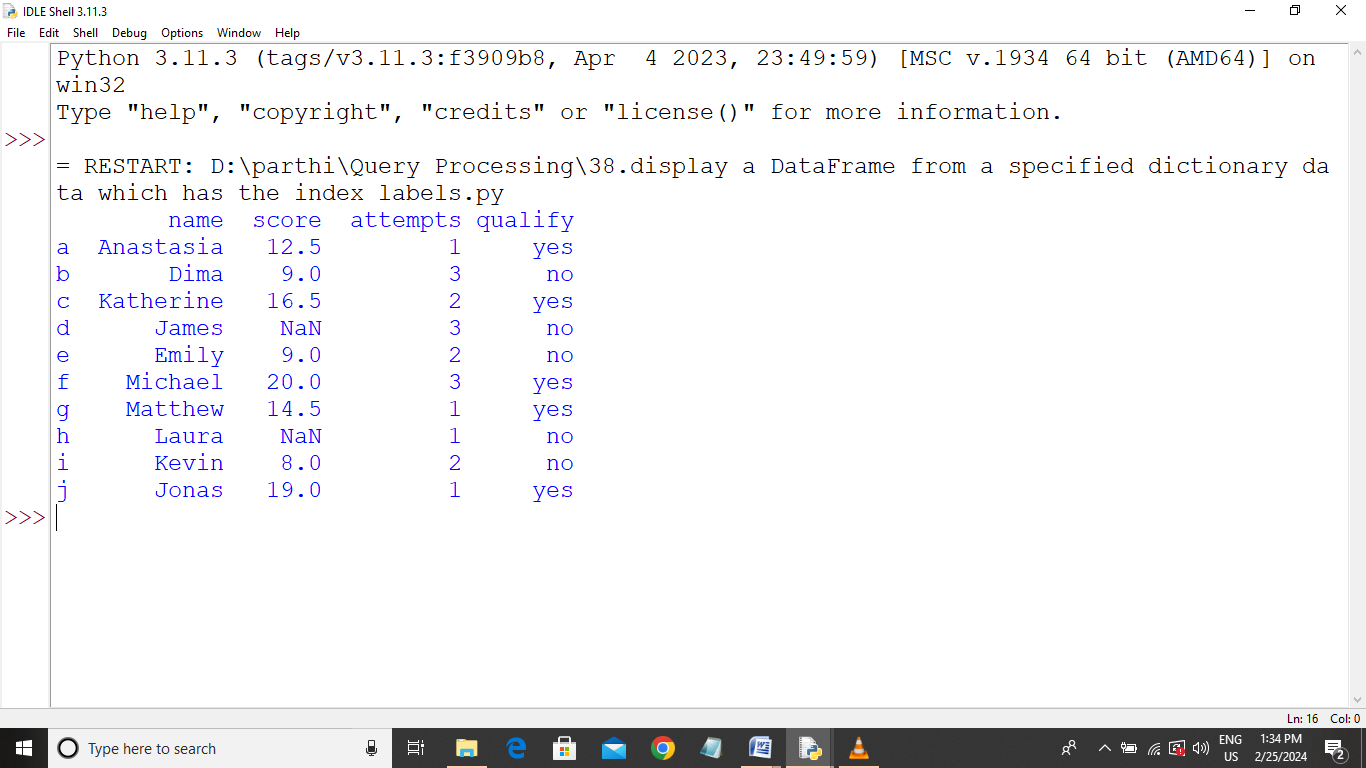
}

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

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

print(df)

**Sample Output:-**



39.Write a Pandas program to get the first 3 rows of a given DataFrame.  
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']

#Parthibhan R

#192224275

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']

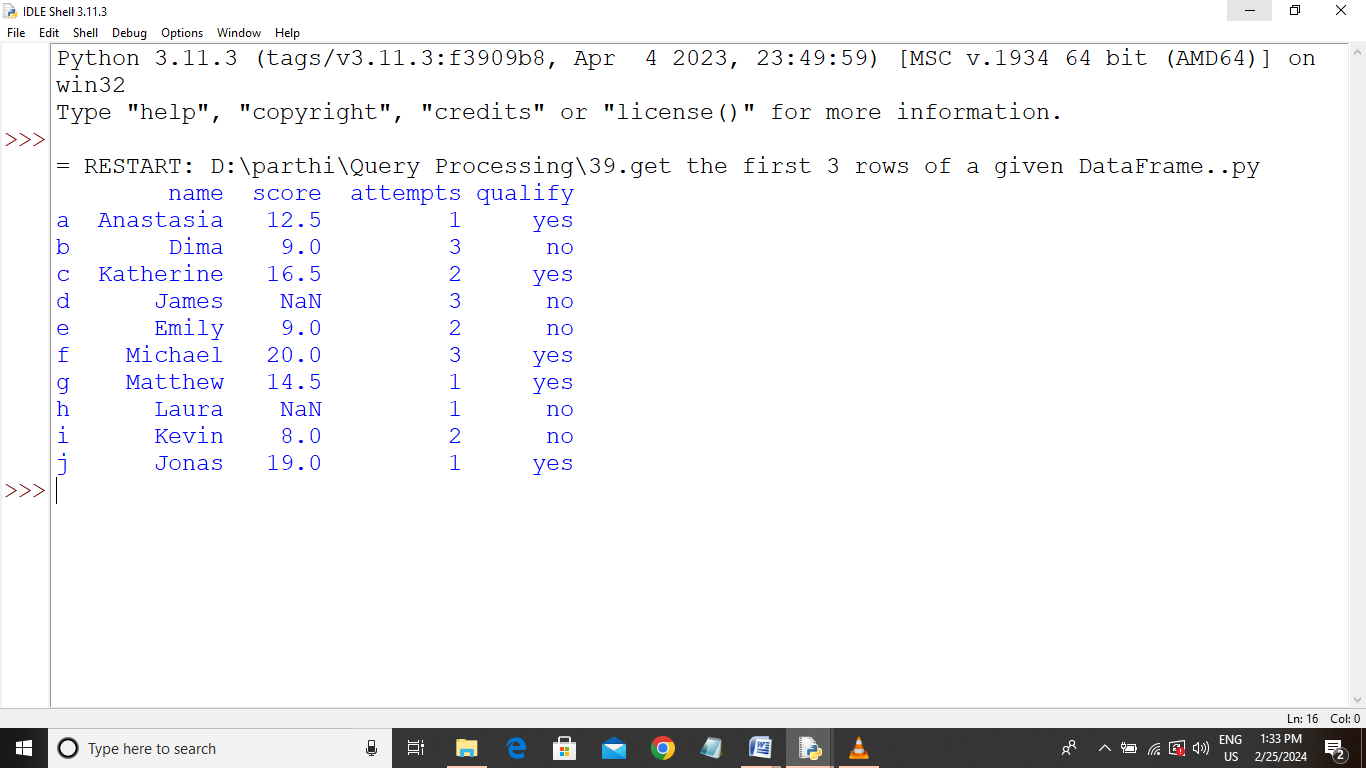
}

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

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

print(df)

**Sample Output:-**

****

40.Write a Pandas program to select the 'name' and 'score' columns from the following DataFrame.  
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']

#Parthibhan R

#192224275

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)

selected\_columns = df[['name', 'score']]

print(selected\_columns)

**Sample Output:-**

