DSA0508 – QUERY PROCESSING FOR DATA SCIENCE WITH DATA EXPLORATION

**NAME : HABEEB MOHAMED KASIM M  
REG NO : 192224093**

**Q1: PROGRAM**

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]

}

employees\_df = pd.DataFrame(data)

distinct\_department\_ids = employees\_df['DEPARTMENT\_ID'].unique() print(distinct\_department\_ids)

**OUTPUT:**

[ 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]

**Q2: PROGRAM**

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]

}

employees\_df = pd.DataFrame(data)

employee\_jobs\_count = employees\_df.groupby('EMPLOYEE\_ID')['JOB\_ID'].nunique() employees\_with\_multiple\_jobs = employee\_jobs\_count[employee\_jobs\_count >= 2] print(employees\_with\_multiple\_jobs.index.tolist())

**OUTPUT:**

[101, 176, 200]

**Q3: PROGRAM**

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]

}

jobs\_df = pd.DataFrame(data)

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

**OUTPUT:**

JOB\_ID JOB\_TITLE MIN\_SALARY MAX\_SALARY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 11 | ST\_MAN | Stock Manager | 5500 | 8500 | |
| 12 | ST\_CLERK | Stock Clerk | 2008 | 5000 | |
| 13 | SH\_CLERK | Shipping Clerk | 2500 | 5500 | |
| 8 | SA\_REP | Sales Representative | 6000 | 12008 | |
| 7 | SA\_MAN | Sales Manager | 10000 | 20080 | |
| 9 | PU\_MAN | Purchasing Manager 8000 | | | 15000 |
| 10 | PU\_CLERK | Purchasing Clerk 2500 | | | 5500 |

18 PR\_REP Public Relations Representative 4500 10500

6 AC\_ACCOUNT Public Accountant 4200 9000

14 IT\_PROG Programmer 4000 10000

0 AD\_PRES President 20080 40000

16 MK\_REP Marketing Representative 4000 9000

15 MK\_MAN Marketing Manager 9000 15000

|  |  |  |
| --- | --- | --- |
| 17 HR\_REP Human Resources Representative | 4000 | 9000 |
| 3 FI\_MGR Finance Manager 8200 | 16000 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | AD\_VP | Administration Vice President | 15000 | 30000 |
| 2 | AD\_ASST | Administration Assistant | 3000 | 6000 |
| 5 | AC\_MGR | Accounting Manager | 8200 | 16000 |

4 FI\_ACCOUNT Accountant 4200 9000

**Q4: PROGRAM**

import pandas as pd

import matplotlib.pyplot as plt import numpy as np start\_date = '2024-01-01'

end\_date = '2024-03-25'

dates = pd.date\_range(start=start\_date, end=end\_date) num\_days = len(dates)

np.random.seed(0)

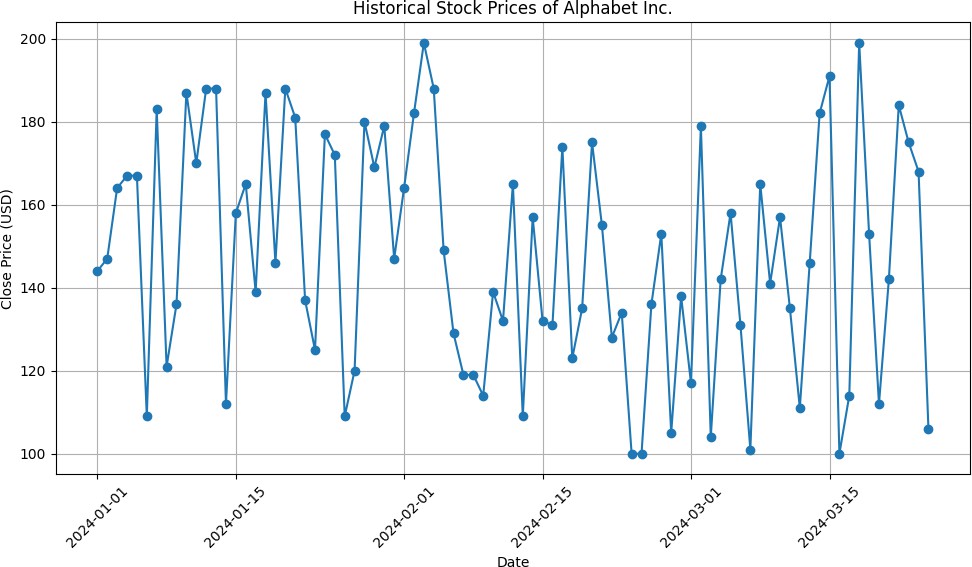
stock\_prices = np.random.randint(100, 200, size=num\_days).astype(float) df = pd.DataFrame({'Date': dates, 'Close': stock\_prices}) plt.figure(figsize=(10, 6))

plt.plot(df['Date'], df['Close'], marker='o', linestyle='-') plt.title('Historical Stock Prices of Alphabet Inc.') plt.xlabel('Date')

plt.ylabel('Close Price (USD)') plt.grid(True)

plt.xticks(rotation=45) plt.tight\_layout() plt.show()

**OUTPUT:**



**Q5: PROGRAM**

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np start\_date = '2024-01-01'

end\_date = '2024-03-25'

dates = pd.date\_range(start=start\_date, end=end\_date) num\_days = len(dates)

np.random.seed(0)

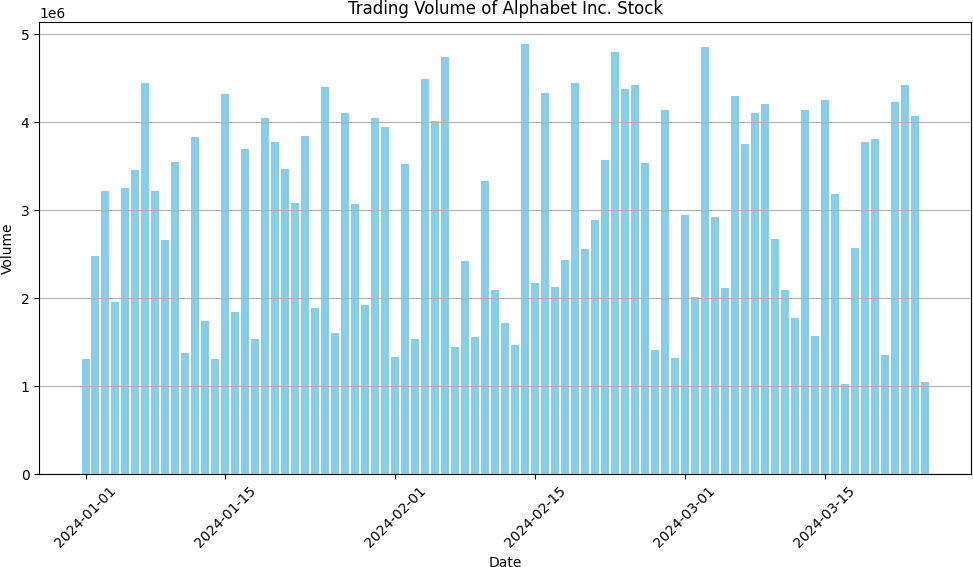
trading\_volume = np.random.randint(1000000, 5000000, size=num\_days) df = pd.DataFrame({'Date': dates, 'Volume': trading\_volume}) plt.figure(figsize=(10, 6))

plt.bar(df['Date'], df['Volume'], color='skyblue') plt.title('Trading Volume of Alphabet Inc. Stock') plt.xlabel('Date')

plt.ylabel('Volume') plt.grid(axis='y')

plt.xticks(rotation=45) plt.tight\_layout() plt.show()

**OUTPUT:**



**Q6: PROGRAM**

import pandas as pd

import matplotlib.pyplot as plt import numpy as np start\_date = '2024-01-01'

end\_date = '2024-03-25'

dates = pd.date\_range(start=start\_date, end=end\_date)

num\_days = len(dates) np.random.seed(0)

trading\_volume = np.random.randint(1000000, 5000000, size=num\_days) stock\_prices = np.random.randint(100, 200, size=num\_days)

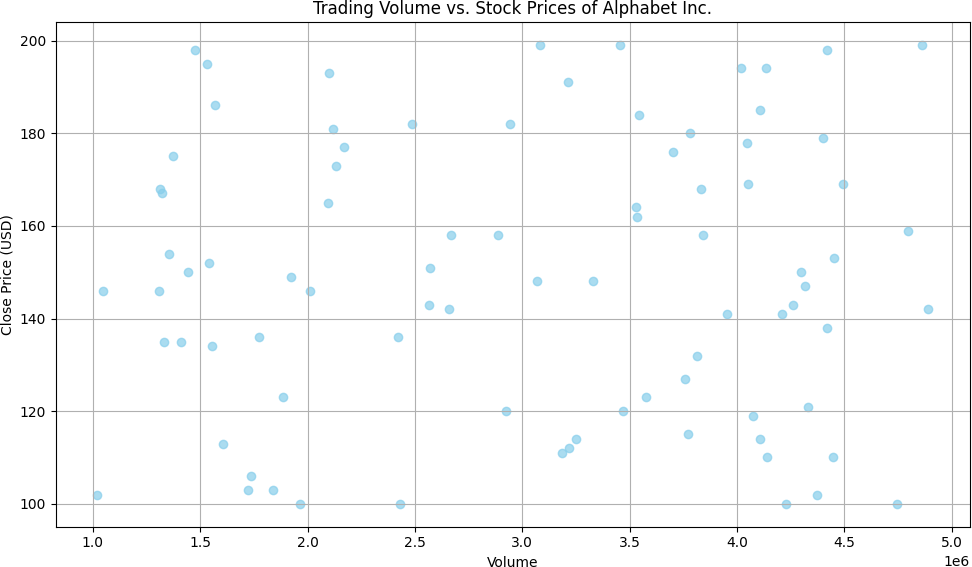
df = pd.DataFrame({'Date': dates, 'Volume': trading\_volume, 'Close': stock\_prices}) plt.figure(figsize=(10, 6))

plt.scatter(df['Volume'], df['Close'], color='skyblue', alpha=0.7) plt.title('Trading Volume vs. Stock Prices of Alphabet Inc.') plt.xlabel('Volume')

plt.ylabel('Close Price (USD)') plt.grid(True)

plt.tight\_layout() plt.show()

**OUTPUT:**



**Q7: PROGRAM**

import pandas as pd sales\_data = {

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

'Date': ['2024-01-01', '2024-01-01', '2024-01-01', '2024-01-02', '2024-01-02', '2024-01-02', '2024-

01-03', '2024-01-03', '2024-01-03'],

'Sale': [100, 150, 200, 120, 170, 220, 130, 180, 230]

}

df = pd.DataFrame(sales\_data)

pivot\_table = pd.pivot\_table(df, values='Sale', index='Item', aggfunc=['max', 'min']) print("Pivot Table:")

print(pivot\_table)

max\_sale\_value = pivot\_table['max'].max().iloc[0] min\_sale\_value = pivot\_table['min'].min().iloc[0] print("\nMaximum Sale Value:", max\_sale\_value) print("Minimum Sale Value:", min\_sale\_value)

**OUTPUT:**

Pivot Table: max min Sale Sale

Item

A 130 100

B 180 150

C 230 200

Maximum Sale Value: 230 Minimum Sale Value: 100

**Q8: PROGRAM**

import pandas as pd sales\_data = {

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

'Date': ['2024-01-01', '2024-01-01', '2024-01-01', '2024-01-02', '2024-01-02', '2024-01-02', '2024-

01-03', '2024-01-03', '2024-01-03'],

'Unit\_Sold': [10, 15, 20, 12, 17, 22, 13, 18, 23]

}

df = pd.DataFrame(sales\_data)

pivot\_table = pd.pivot\_table(df, values='Unit\_Sold', index='Item', aggfunc='sum') print("Pivot Table - Item wise unit sold:")

print(pivot\_table)

**OUTPUT:**

Pivot Table - Item wise unit sold: Unit\_Sold

Item

|  |  |
| --- | --- |
| A | 35 |
| B | 50 |
| C | 65 |

**Q9: PROGRAM**

import pandas as pd sales\_data = {

'Region': ['East', 'East', 'West', 'West', 'North', 'North', 'South', 'South', 'East', 'West'],

'Manager': ['John', 'John', 'Smith', 'Smith', 'Emma', 'Emma', 'Adam', 'Adam', 'John', 'Smith'],

'Salesman': ['Alex', 'Bob', 'Charlie', 'David', 'Ethan', 'Frank', 'George', 'Harry', 'Ian', 'Jack'], 'Sale\_Amount': [1000, 1500, 1200, 1700, 1300, 1800, 1400, 1900, 1600, 1100]

}

df = pd.DataFrame(sales\_data)

pivot\_table = pd.pivot\_table(df, values='Sale\_Amount', index=['Region', 'Manager', 'Salesman'], aggfunc='sum')

print("Pivot Table - Total Sale Amount (Region-wise, Manager-wise, Salesman-wise):") print(pivot\_table)

**OUTPUT:**

Pivot Table - Total Sale Amount (Region-wise, Manager-wise, Salesman-wise): Sale\_Amount

Region Manager Salesman

|  |  |  |  |
| --- | --- | --- | --- |
| East John Alex | | 1000 | |
| Bob | | 1500 | |
| Ian | | 1600 | |
| North Emma | Ethan | | 1300 |
| Frank | 1800 | |  |
| South Adam | George | | 1400 |
| Harry | 1900 | |  |

West Smith Charlie 1200

David 1700

Jack 1100

**Q10: PROGRAM**

import pandas as pd import numpy as np np.random.seed(0)

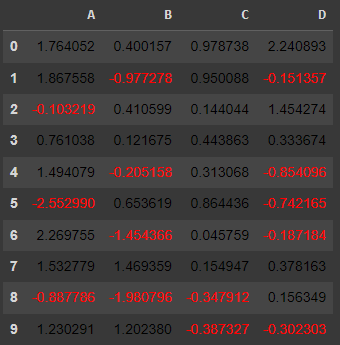
data = np.random.randn(10, 4)

df = pd.DataFrame(data, columns=['A', 'B', 'C', 'D']) def color\_negative\_red(val):

color = 'red' if val < 0 else 'black' return f'color: {color}'

styled\_df = df.style.applymap(color\_negative\_red) styled\_df

**OUTPUT:**



**Q11: PROGRAM**

import pandas as pd import numpy as np np.random.seed(0)

data = np.random.randn(10, 5)

df = pd.DataFrame(data, columns=['A', 'B', 'C', 'D', 'E']) df.loc[4:4, 'B'] = np.nan

df.loc[3:3, 'D'] = np.nan

df.loc[0:0, 'C'] = np.nan

df.loc[9:9, 'E'] = np.nan def highlight\_nan(val):

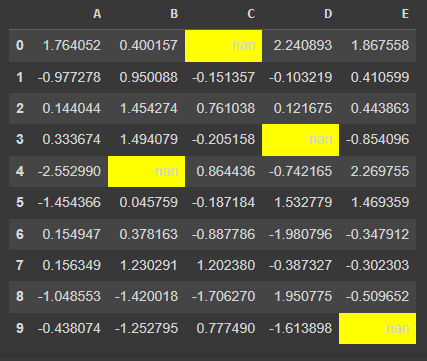
if pd.isna(val):

return 'background-color: yellow' else:

return ''

styled\_df = df.style.applymap(highlight\_nan) styled\_df

**OUTPUT:**



**Q12: PROGRAM**

import pandas as pd import numpy as np np.random.seed(0)

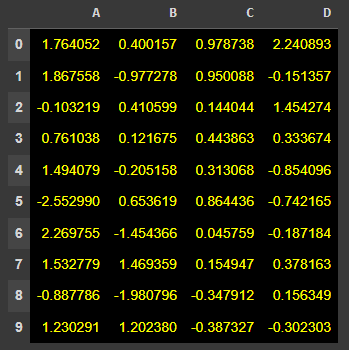
data = np.random.randn(10, 4)

df = pd.DataFrame(data, columns=['A', 'B', 'C', 'D']) def set\_colors(val):

return 'background-color: black; color: yellow'

styled\_df = df.style.applymap(lambda x: set\_colors(x)) styled\_df

**OUTPUT:**



**Q13: PROGRAM**

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],

'purch\_amt': [None, 150.50, None, None, 65.26, 110.50, 270.65, 1983.43, 2480.40, 250.45, 75.29],

'ord\_date': ['2012-10-05', '2012-09-10', None, '2012-09-10', '2012-08-17', '2012-09-10', '2012-07-

27', '2012-09-10', '2012-10-10', '2012-06-27', '2012-08-17'],

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

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

5003.0]

}

df = pd.DataFrame(data) missing\_values = df.isna() print(missing\_values)

**OUTPUT:**

ord\_no purch\_amt ord\_date customer\_id salesman\_id

1. False True False False False
2. True False False False False
3. False True True False False
4. False True False False True
5. True False False False False
6. False False False False False
7. True False False False False
8. False False False False True

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 8 False | False | False | False | False |
| 9 False | False | False | False | False |
| 10 True | False | False | False | False |

**Q14: PROGRAM**

import pandas as pd import numpy as np data = {

'ord\_no': [70001, 70002, 70004, np.nan, 70005, 5760, 70010, 70003, 70012, np.nan, 70013],

'purch\_amt': [1, 65.26, 110.5, 948.5, 2400.6, 5760, '?', 12.43, 2480.4, 250.45, 3045.6],

'ord\_date': ['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, 3003, 3002, 3001, 3001, 3004, 3003, 3002, 3001, np.nan],

'salesman\_id': [5002, np.nan, 5001, np.nan, 5002, 5001, np.nan, 5003, 5002, 5003, np.nan]

}

df = pd.DataFrame(data)

df.replace('?', np.nan, inplace=True) df.fillna(0, inplace=True)

print(df)

**OUTPUT:**

ord\_no purch\_amt ord\_date customer\_id salesman\_id

|  |  |  |
| --- | --- | --- |
| 0 70001.0 | 1.00 2012-09-10 3002.0 | 5002.0 |
| 1 70002.0 | 65.26 0 3001.0 | 0.0 |
| 2 70004.0 | 110.50 2012-08-17 3003.0 | 5001.0 |

3 0.0 948.50 2012-09-10 3002.0 0.0

4 70005.0 2400.60 2012-07-27 3001.0 5002.0

5 5760.0 5760.00 2012-09-10 3001.0 5001.0

6 70010.0 0.00 2012-10-10 3004.0 0.0

7 70003.0 12.43 2012-10-10 3003.0 5003.0

1. 70012.0 2480.40 2012-06-27 3002.0 5002.0

9 0.0 250.45 2012-08-17 3001.0 5003.0

10 70013.0 3045.60 2012-04-25 0.0 0.0

**Q15: PROGRAM**

import pandas as pd import numpy as np data = {

'ord\_no': [np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan],

'purch\_amt': [1, 2, np.nan, 4, 5, 6, np.nan, np.nan, 9, 10, 11],

'ord\_date': [np.nan, np.nan, 3, np.nan, np.nan, np.nan, 7, 8, 9, np.nan, 11],

'customer\_id': [np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan, np.nan]

}

df = pd.DataFrame(data) print(df)

**OUTPUT:**

ord\_no purch\_amt ord\_date customer\_id

1. NaN 1.0 NaN NaN
2. NaN 2.0 NaN NaN
3. NaN NaN 3.0 NaN
4. NaN 4.0 NaN NaN
5. NaN 5.0 NaN NaN
6. NaN 6.0 NaN NaN
7. NaN NaN 7.0 NaN
8. NaN NaN 8.0 NaN
9. NaN 9.0 9.0 NaN
10. NaN 10.0 NaN NaN
11. NaN 11.0 11.0 NaN

**Q16: PROGRAM**

import pandas as pd data = {

'school': ['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']

}

df = pd.DataFrame(data) grouped = df.groupby('school') print(type(grouped))

for name, group in grouped: print("\nSchool Code:", name) print(group)

**OUTPUT:**

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

School Code: s001

school class name date\_of\_birth age height weight address

0 s001 V Alberto Franco 15/05/2002 12 173 35 street1

3 s001 VI Eesha Hinton 25/09/1998 13 167 30 street1

School Code: s002

school class name date\_of\_birth age height weight address

|  |  |  |  |
| --- | --- | --- | --- |
| 1 s002 | V Gino Mcneill 17/05/2002 12 | 192 | 32 street2 |
| 4 s002 | V Gino Mcneill 11/05/2002 14 | 151 | 31 street2 |

School Code: s003

school class name date\_of\_birth age height weight address

2 s003 VI Ryan Parkes 16/02/1999 13 186 33 street3

School Code: s004

school class name date\_of\_birth age height weight address

5 s004 VI David Parkes 15/09/1997 12 159 32 street4

**Q17: PROGRAM**

import pandas as pd data = {

'school': ['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']

}

df = pd.DataFrame(data)

result = df.groupby('school')['age'].agg(['mean', 'min', 'max']) print("Mean, Min, and Max Age for Each School:")

print(result)

**OUTPUT:**

Mean, Min, and Max Age for Each School: mean min max

school

|  |  |  |  |
| --- | --- | --- | --- |
| s001 | 12.5 | 12 | 13 |
| s002 | 13.0 | 12 | 14 |
| s003 | 13.0 | 13 | 13 |
| s004 | 12.0 | 12 | 12 |

**Q18: PROGRAM**

import pandas as pd data = {

'school': ['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']

}

df = pd.DataFrame(data)

grouped = df.groupby(['school', 'class']) for name, group in grouped:

print("\nGroup:", name) print(group)

**OUTPUT:**

Group: ('s001', 'V')

school class name date\_of\_birth age height weight address

0 s001 V Alberto Franco 15/05/2002 12 173 35 street1

Group: ('s001', 'VI')

school class name date\_of\_birth age height weight address

3 s001 VI Eesha Hinton 25/09/1998 13 167 30 street1

Group: ('s002', 'V')

school class name date\_of\_birth age height weight address

|  |  |  |  |
| --- | --- | --- | --- |
| 1 s002 | V Gino Mcneill 17/05/2002 12 | 192 | 32 street2 |
| 4 s002 | V Gino Mcneill 11/05/2002 14 | 151 | 31 street2 |

Group: ('s003', 'VI')

school class name date\_of\_birth age height weight address

2 s003 VI Ryan Parkes 16/02/1999 13 186 33 street3

Group: ('s004', 'VI')

school class name date\_of\_birth age height weight address

5 s004 VI David Parkes 15/09/1997 12 159 32 street4

**Q19: PROGRAM**

import pandas as pd data = {

'Year': [1986, 1986, 1985, 1986, 1987],

'WHO region': ['Western Pacific', 'Americas', 'Africa', 'Americas', 'Americas'], 'Country': ['Viet Nam', 'Uruguay', "Cte d'Ivoire", 'Colombia', 'Saint Kitts and Nevis'], 'Beverage Types': ['Wine', 'Other', 'Wine', 'Beer', 'Beer'],

'Display Value': [0.00, 0.50, 1.62, 4.27, 1.98]

}

df = pd.DataFrame(data)

print("Dimensions or Shape of the DataFrame:", df.shape) column\_names = df.columns.tolist()

print("Column Names:")

for name in column\_names: print(name)

**OUTPUT:**

Dimensions or Shape of the DataFrame: (5, 5) Column Names:

Year

WHO region Country

Beverage Types Display Value

**Q20: PROGRAM**

import pandas as pd

data = {'Text': ['apple', 'banana', 'orange', 'grape']} df = pd.DataFrame(data)

def find\_substring\_index(dataframe, column\_name, substring): indices = []

for index, row in dataframe.iterrows():

index\_of\_substring = row[column\_name].find(substring) indices.append(index\_of\_substring)

return indices substring = 'ra'

df['Substring\_Index'] = find\_substring\_index(df, 'Text', substring) print(df)

**OUTPUT:**

Text Substring\_Index

|  |  |
| --- | --- |
| 0 apple | -1 |
| 1 banana | -1 |
| 2 orange | 1 |
| 3 grape | 1 |

**Q21: PROGRAM**

import pandas as pd

data = {'Name': ['John', 'Alice', 'Bob', 'Diana'], 'Age': [25, 30, 35, 40]}

df = pd.DataFrame(data)

df['Name'] = df['Name'].str.swapcase() print(df)

**OUTPUT:**

Name Age

1. jOHN 25
2. aLICE 30
3. bOB 35
4. dIANA 40

**Q22: PROGRAM**

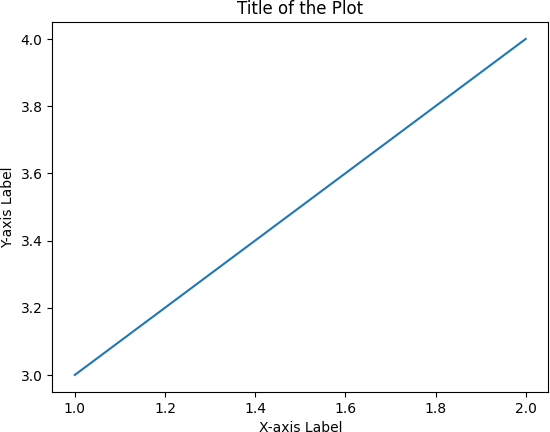
import matplotlib.pyplot as plt x = [1,2]

y = [3,4]

plt.plot(x, y)

plt.xlabel('X-axis Label') plt.ylabel('Y-axis Label') plt.title('Title of the Plot') plt.show()

**OUTPUT:**



**Q23: PROGRAM**

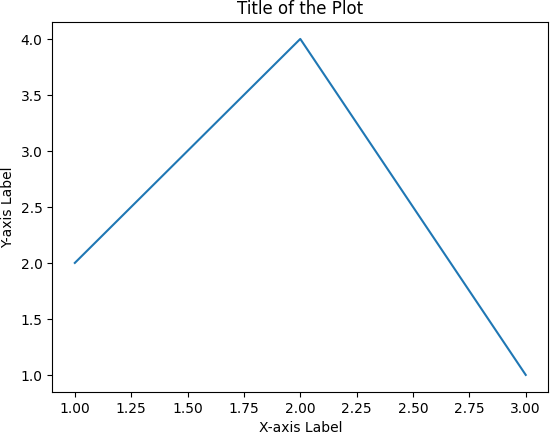
import matplotlib.pyplot as plt x = [1,2,3]

y = [2,4,1]

plt.plot(x, y)

plt.xlabel('X-axis Label') plt.ylabel('Y-axis Label') plt.title('Title of the Plot') plt.show()

**OUTPUT:**



**Q24: PROGRAM**

import pandas as pd

import matplotlib.pyplot as plt financial\_data = {

'Date': ['10-03-16', '10-04-16', '10-05-16', '10-06-16', '10-07-16'],

'Open': [774.25, 776.030029, 779.309998, 779, 779.659973],

'High': [776.065002, 778.710022, 782.070007, 780.47998, 779.659973],

'Low': [769.5, 772.890015, 775.650024, 775.539978, 770.75],

'Close': [772.559998, 776.429993, 776.469971, 776.859985, 775.080017]

}

df = pd.DataFrame(financial\_data)

df['Date'] = pd.to\_datetime(df['Date'], format='%m-%d-%y') plt.figure(figsize=(10, 6))

plt.plot(df['Date'], df['Open'], label='Open')

plt.plot(df['Date'], df['High'], label='High')

plt.plot(df['Date'], df['Low'], label='Low')

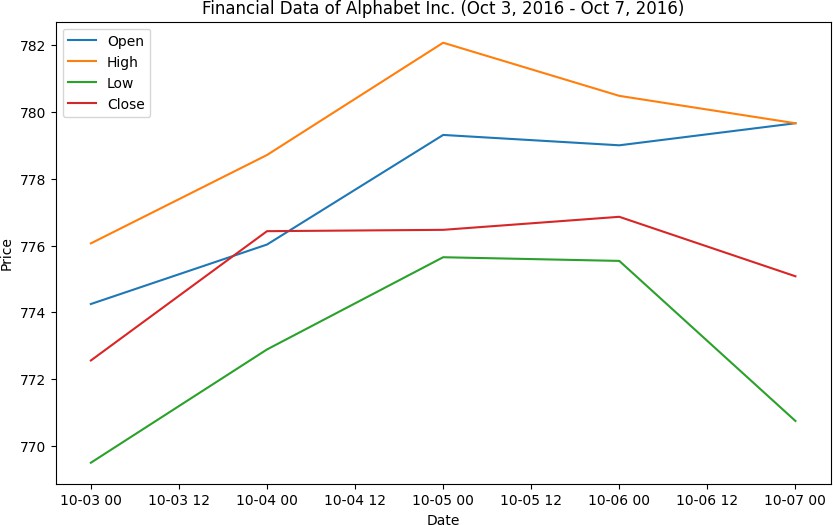
plt.plot(df['Date'], df['Close'], label='Close') plt.xlabel('Date')

plt.ylabel('Price')

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

plt.show()

**OUTPUT:**



**Q25: PROGRAM**

import matplotlib.pyplot as plt x = [1, 2, 3, 4, 5]

y1 = [2, 3, 5, 7, 11]

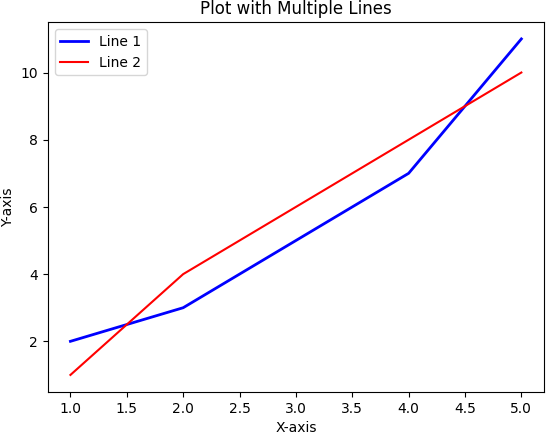
y2 = [1, 4, 6, 8, 10]

plt.plot(x, y1, label='Line 1', color='blue', linewidth=2) plt.plot(x, y2, label='Line 2', color='red', linewidth=1.5) plt.legend()

plt.xlabel('X-axis') plt.ylabel('Y-axis')

plt.title('Plot with Multiple Lines') plt.show()

**OUTPUT:**



**Q26: PROGRAM**

import matplotlib.pyplot as plt import numpy as np

x = np.linspace(0, 10, 100) y1 = np.sin(x)

y2 = np.cos(x)

fig, axs = plt.subplots(2) axs[0].plot(x, y1, color='blue') axs[0].set\_title('Sin(x)')

axs[1].plot(x, y2, color='red')

axs[1].set\_title('Cos(x)') plt.show()

plt.figure(1)

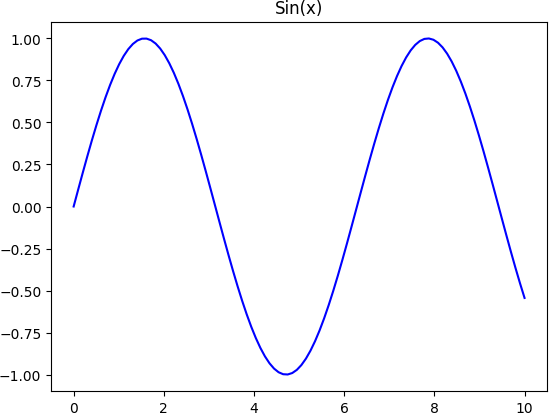
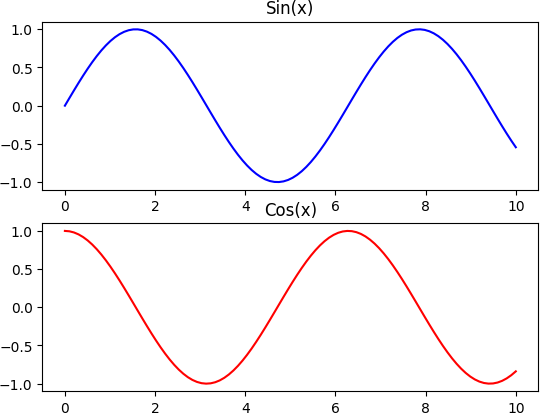
plt.plot(x, y1, color='blue') plt.title('Sin(x)')

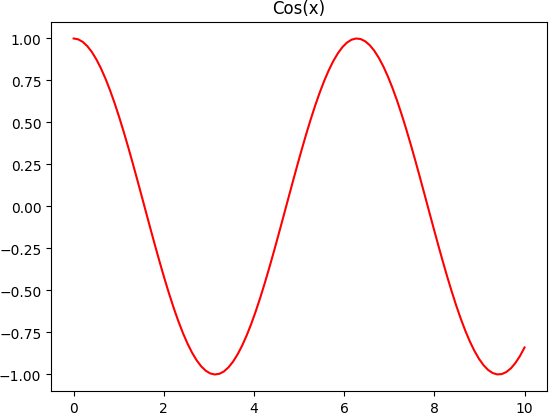
plt.show()

plt.figure(2)

plt.plot(x, y2, color='red') plt.title('Cos(x)') plt.show()

**OUTPUT:**





**Q27: PROGRAM**

import matplotlib.pyplot as plt

languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'] popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

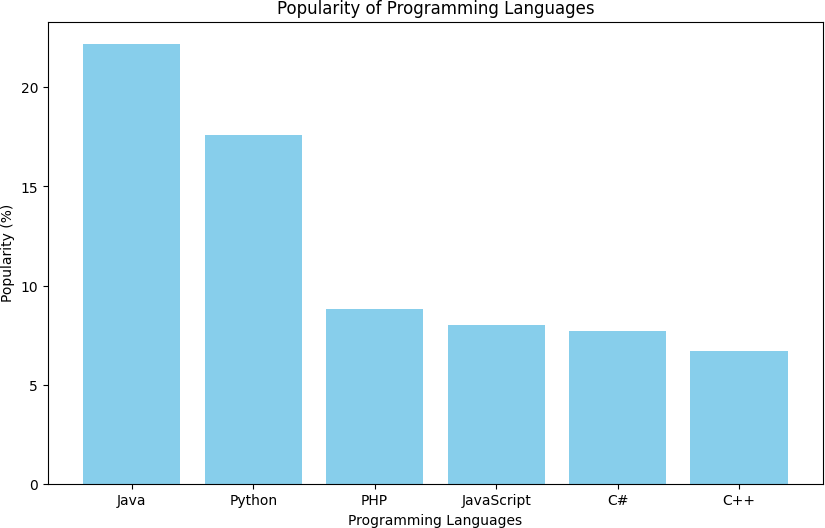
plt.figure(figsize=(10, 6))

plt.bar(languages, popularity, color='skyblue') plt.xlabel('Programming Languages')

plt.ylabel('Popularity (%)')

plt.title('Popularity of Programming Languages') plt.show()

**OUTPUT:**



**Q28: PROGRAM**

import matplotlib.pyplot as plt

languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'] popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

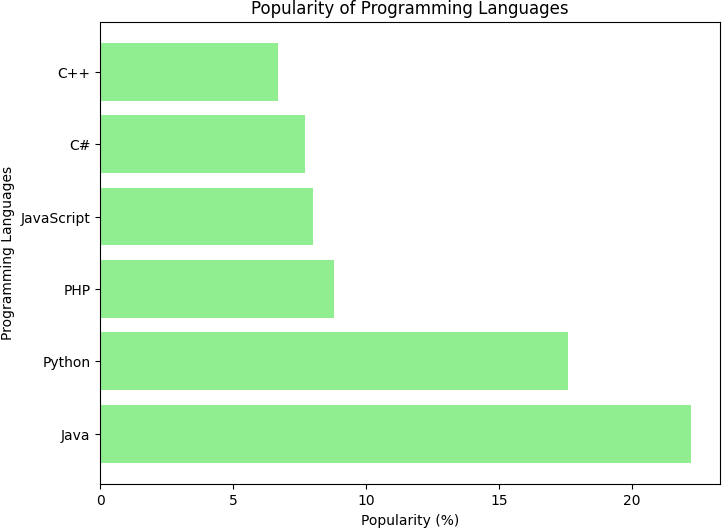
plt.figure(figsize=(8, 6))

plt.barh(languages, popularity, color='lightgreen') plt.xlabel('Popularity (%)')

plt.ylabel('Programming Languages')

plt.title('Popularity of Programming Languages') plt.show()

**OUTPUT:**



**Q29: PROGRAM**

import matplotlib.pyplot as plt

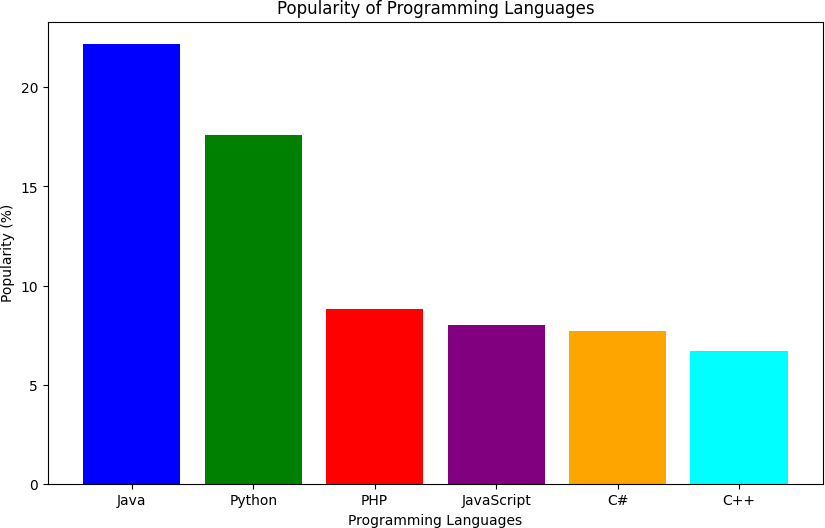
languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'] popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

colors = ['blue', 'green', 'red', 'purple', 'orange', 'cyan'] plt.figure(figsize=(10, 6))

plt.bar(languages, popularity, color=colors) plt.xlabel('Programming Languages') plt.ylabel('Popularity (%)')

plt.title('Popularity of Programming Languages') plt.show()

**OUTPUT:**



**Q30: PROGRAM**

import numpy as np

import matplotlib.pyplot as plt men\_means = (22, 30, 35, 35, 26)

women\_means = (25, 32, 30, 35, 29)

group\_labels = ['Group 1', 'Group 2', 'Group 3', 'Group 4', 'Group 5']

bar\_width = 0.35

index = np.arange(len(group\_labels)) plt.figure(figsize=(10, 6))

plt.bar(index, men\_means, bar\_width, label='Men')

plt.bar(index + bar\_width, women\_means, bar\_width, label='Women') plt.xlabel('Groups')

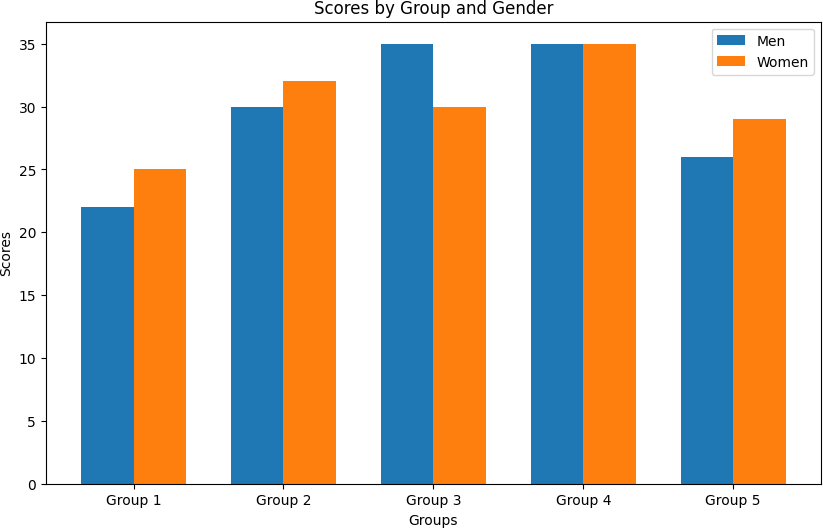
plt.ylabel('Scores')

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

plt.xticks(index + bar\_width / 2, group\_labels) plt.legend()

plt.show()

**OUTPUT:**



**Q31: PROGRAM**

import numpy as np

import matplotlib.pyplot as plt men\_means = (22, 30, 35, 35, 26)

women\_means = (25, 32, 30, 35, 29)

men\_std = (4, 3, 4, 1, 5)

women\_std = (3, 5, 2, 3, 3)

group\_labels = ['Group 1', 'Group 2', 'Group 3', 'Group 4', 'Group 5'] bottom\_positions = np.array(men\_means)

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

bars1 = plt.bar(range(len(group\_labels)), men\_means, yerr=men\_std, label='Men') bars2 = plt.bar(range(len(group\_labels)), women\_means, yerr=women\_std, bottom=bottom\_positions, label='Women')

plt.xlabel('Groups') plt.ylabel('Scores')

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

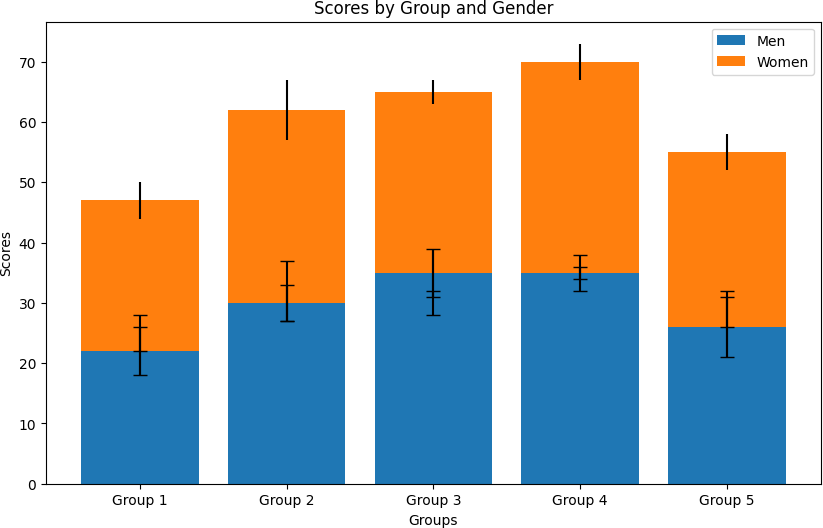
plt.xticks(range(len(group\_labels)), group\_labels) plt.legend()

plt.errorbar(range(len(group\_labels)), men\_means, yerr=men\_std, fmt='none', ecolor='black', capsize=5)

plt.errorbar(range(len(group\_labels)), women\_means, yerr=women\_std, fmt='none', ecolor='black', capsize=5)

plt.show()

**OUTPUT:**



**Q32: PROGRAM**

import numpy as np

import matplotlib.pyplot as plt np.random.seed(0)

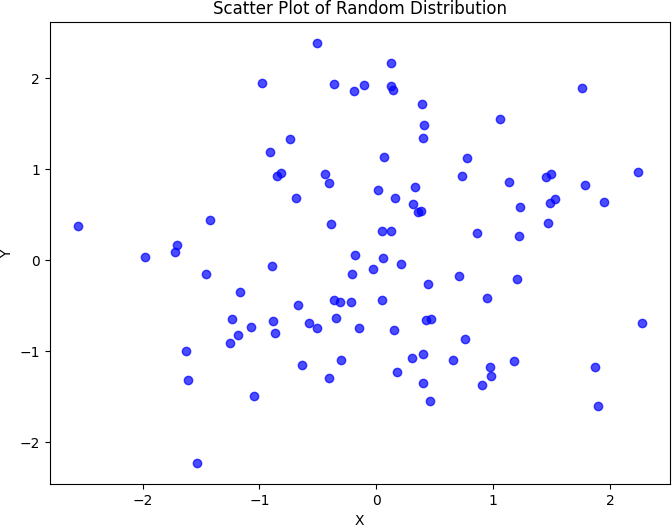
x = np.random.randn(100) y = np.random.randn(100) plt.figure(figsize=(8, 6))

plt.scatter(x, y, color='blue', alpha=0.7) plt.xlabel('X')

plt.ylabel('Y')

plt.title('Scatter Plot of Random Distribution') plt.show()

**OUTPUT:**



**Q33: PROGRAM**

import numpy as np

import matplotlib.pyplot as plt np.random.seed(0)

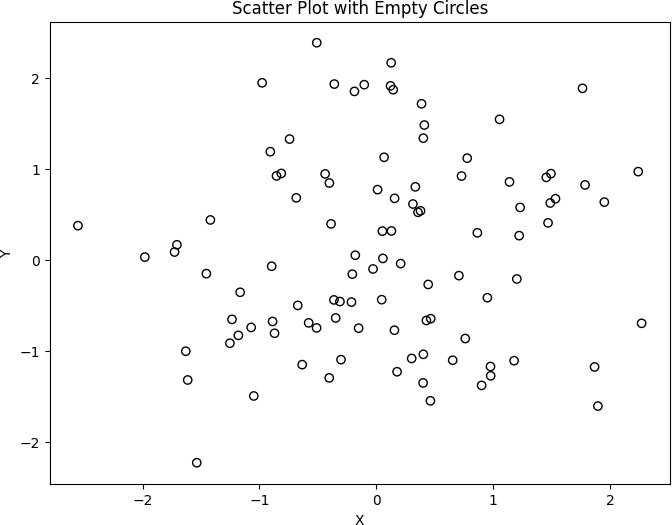
x = np.random.randn(100) y = np.random.randn(100) plt.figure(figsize=(8, 6))

plt.scatter(x, y, color='blue', edgecolor='black', facecolor='none') plt.xlabel('X')

plt.ylabel('Y')

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

**OUTPUT:**



**Q34: PROGRAM**

import numpy as np

import matplotlib.pyplot as plt np.random.seed(0)

x = np.random.rand(100) y = np.random.rand(100)

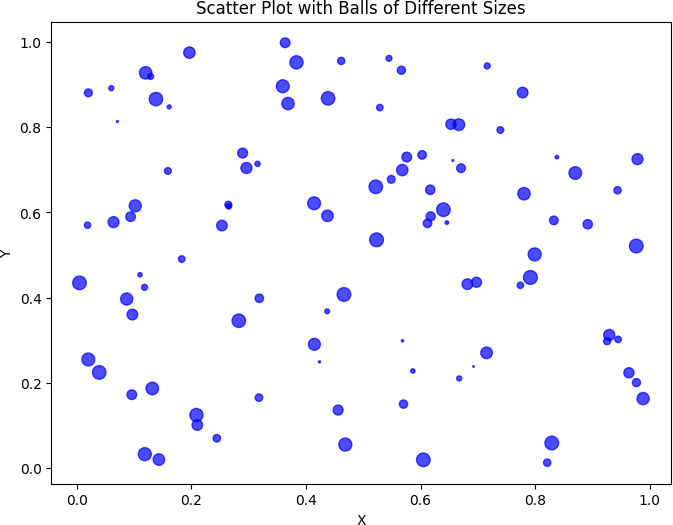
sizes = np.random.rand(100) \* 100 plt.figure(figsize=(8, 6))

plt.scatter(x, y, s=sizes, color='blue', alpha=0.7) plt.xlabel('X')

plt.ylabel('Y')

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

**OUTPUT:**



**Q35: PROGRAM**

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]

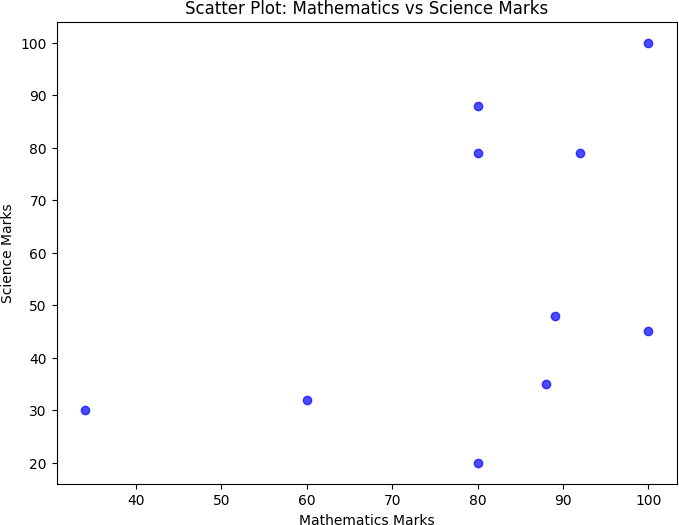
plt.figure(figsize=(8, 6))

plt.scatter(math\_marks, science\_marks, color='blue', alpha=0.7) plt.xlabel('Mathematics Marks')

plt.ylabel('Science Marks')

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

**OUTPUT:**



**Q36: PROGRAM**

import matplotlib.pyplot as plt import numpy as np

np.random.seed(0) num\_samples = 50

heights\_group1 = np.random.normal(loc=160, scale=10, size=num\_samples) weights\_group1 = np.random.normal(loc=60, scale=10, size=num\_samples) heights\_group2 = np.random.normal(loc=170, scale=12, size=num\_samples) weights\_group2 = np.random.normal(loc=70, scale=12, size=num\_samples) heights\_group3 = np.random.normal(loc=165, scale=8, size=num\_samples) weights\_group3 = np.random.normal(loc=65, scale=8, size=num\_samples) plt.figure(figsize=(10, 6))

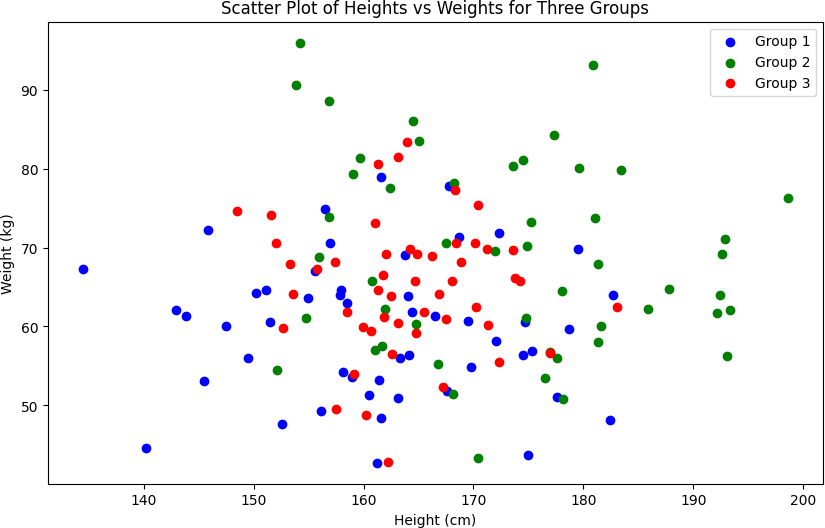
plt.scatter(heights\_group1, weights\_group1, color='blue', label='Group 1') plt.scatter(heights\_group2, weights\_group2, color='green', label='Group 2') plt.scatter(heights\_group3, weights\_group3, color='red', label='Group 3') plt.xlabel('Height (cm)')

plt.ylabel('Weight (kg)')

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

plt.show()

**OUTPUT:**



**Q37: PROGRAM**

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)

**OUTPUT:**

X Y Z

0 78 84 86

1 85 94 97

2 96 89 96

3 80 83 72

1. 86 86 83

**Q38: PROGRAM**

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 | |

**Q39: PROGRAM**

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) first\_3\_rows = df.head(3)

print("First 3 rows of the DataFrame:") print(first\_3\_rows)

**OUTPUT:**

First 3 rows of the DataFrame:

name score attempts qualify

a Anastasia 12.5 1 yes

b Dima 9.0 3 no

c Katherine 16.5 2 yes

**Q40: PROGRAM**

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)

**OUTPUT:**

name score

a Anastasia 12.5

b Dima 9.0

c Katherine 16.5

1. James NaN
2. Emily 9.0
3. Michael 20.0
4. Matthew 14.5
5. Laura NaN
6. Kevin 8.0
7. Jonas 19.0