```
   1. import pandas as pd
employees = pd.read_csv(r"EMPLOYEES.csv")
departments = pd.read_csv(r"DEPARTMENTS.csv")
job_history = pd.read_csv(r"JOB_HISTORY.csv")
jobs = pd.read_csv(r"JOBS.csv")
countries = pd.read_csv(r"COUNTRIES.csv")
regions = pd.read_csv(r"REGIONS.csv")
locations = pd.read_csv(r"LOCATIONS.csv")
print("Distinct department_id:")
print(employees.department_id.unique())
```

```
   2. import pandas as pd
employees = pd.read_csv(r"EMPLOYEES.csv")
departments = pd.read_csv(r"DEPARTMENTS.csv")
job_history = pd.read_csv(r"JOB_HISTORY.csv")
jobs = pd.read_csv(r"JOBS.csv")
countries = pd.read_csv(r"COUNTRIES.csv")
regions = pd.read_csv(r"REGIONS.csv")
locations = pd.read_csv(r"LOCATIONS.csv")
result = job_history.groupby(['employee_id'])
print(result.filter(lambda x: len(x)
>1).groupby('employee_id').size().sort_values(ascending=False))
```

```
3.import pandas as pd
employees = pd.read_csv(r"EMPLOYEES.csv")
departments = pd.read_csv(r"DEPARTMENTS.csv")
job_history = pd.read_csv(r"JOB_HISTORY.csv")
jobs = pd.read_csv(r"JOBS.csv")
countries = pd.read_csv(r"COUNTRIES.csv")
regions = pd.read_csv(r"REGIONS.csv")
locations = pd.read_csv(r"LOCATIONS.csv")
print("job_id      Job ID     min_salary  max_salary")
result = jobs.sort_values('job_title')
for index, row in result.iterrows():
print(row['job_id'].ljust(15),row['job_title'].ljust(35),str(row['mi
n_salary']).ljust(9),row['max_salary'])
```

```
4.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()
```

5.
```
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.set_index('Date')
plt.figure(figsize=(6,6))
plt.suptitle('Trading Volume of Alphabet Inc. stock,\n01-04-2020 to
30-04-2020', fontsize=16, color='black')
plt.xlabel("Date",fontsize=12, color='black')
plt.ylabel("Trading Volume", fontsize=12, color='black')
df2['Volume'].plot(kind='bar');
plt.show()
```

6.
```
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.set_index('Date')
x= ['Close']; y = ['Volume']
plt.figure(figsize=[15,10])
df2.plot.scatter(x, y, s=50);
plt.grid(True)
plt.title('Trading Volume/Price of Alphabet Inc. stock,\n01-04-2020
to 30-09-2020', fontsize=14, color='black')
plt.xlabel("Stock Price",fontsize=12, color='black')
plt.ylabel("Trading Volume", fontsize=12, color='black')
```

```
plt.show()
```

7.
```
import pandas as pd
import numpy as np
df = pd.read_excel('E:\SaleData.xlsx')
table = pd.pivot_table(df, index='Item', values='Sale_amt',
aggfunc=[np.max, np.min])
print(table)
```

8.
```
import pandas as pd
import numpy as np
df = pd.read_excel('E:\SaleData.xlsx')
print(pd.pivot_table(df,index=["Item"], values="Units",
aggfunc=np.sum))
```

9.
```
import numpy as np
import pandas as pd
df = pd.read_excel('E:\SaleData.xlsx')
print(pd.pivot_table(df,index=["Region","Manager","SalesMan"],
values="Sale_amt", aggfunc=np.sum))
```

10.
```
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)
```

11.
```
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 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.highlight_null(null_color='red')
```

12.
```
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)
print("\nBackground:black - fontcolor:yelow")
df.style.set_properties(**{'background-color': 'black',
                           'color': 'yellow'})
```

13.
```
import pandas as pd
import numpy as np
pd.set_option('display.max_rows', None)
#pd.set_option('display.max_columns', None)
df = pd.DataFrame({
'ord_no':[70001,np.nan,70002,70004,np.nan,70005,np.nan,70010,70003,7
0012,np.nan,70013],
'purch_amt':[150.5,270.65,65.26,110.5,948.5,2400.6,5760,1983.43,2480
.4,250.45, 75.29,3045.6],
```

```
'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':[3002,3001,3001,3003,3002,3001,3001,3004,3003,3002,300
1,3001],
'salesman_id':[5002,5003,5001,np.nan,5002,5001,5001,np.nan,5003,5002
,5003,np.nan]})
print("Original Orders DataFrame:")
print(df)
print("\nMissing values of the said dataframe:")
print(df.isna())
```

14.
```
import pandas as pd
import numpy as np
pd.set_option('display.max_rows', None)
#pd.set_option('display.max_columns', None)
df = pd.DataFrame({
'ord_no':[70001,np.nan,70002,70004,np.nan,70005,"--
",70010,70003,70012,np.nan,70013],
'purch_amt':[150.5,270.65,65.26,110.5,948.5,2400.6,5760,"?",12.43,24
80.4,250.45, 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':[3002,3001,3001,3003,3002,3001,3001,3004,"--
",3002,3001,3001],
'salesman_id':[5002,5003,"?",5001,np.nan,5002,5001,"?",5003,5002,500
3,"--"]})
print("Original Orders DataFrame:")
print(df)
print("\nReplace the missing values with NaN:")
result = df.replace({"?": np.nan, "--": np.nan})
print(result)
```

15.
```
import pandas as pd
import numpy as np
pd.set_option('display.max_rows', None)
#pd.set_option('display.max_columns', None)
df = pd.DataFrame({
'ord_no':[np.nan,np.nan,70002,np.nan,np.nan,70005,np.nan,70010,70003
,70012,np.nan,np.nan],
'purch_amt':[np.nan,270.65,65.26,np.nan,948.5,2400.6,5760,1983.43,24
80.4,250.45, 75.29,np.nan],
```

```
'ord_date': [np.nan,'2012-09-10',np.nan,np.nan,'2012-09-10','2012-
07-27','2012-09-10','2012-10-10','2012-10-10','2012-06-27','2012-08-
17',np.nan],
'customer_id':[np.nan,3001,3001,np.nan,3002,3001,3001,3004,3003,3002
,3001,np.nan]})
print("Original Orders DataFrame:")
print(df)
print("\nKeep the rows with at least 2 NaN values of the said
DataFrame:")
result = df.dropna(thresh=2)
print(result)
```

```
16.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','1
5/09/1997'],
    '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))
```

```
17. 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','1
5/09/1997'],
    '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 school with
customized column names:')
grouped_single = student_data.groupby('school_code').agg(Age_Mean =
('age','mean'),Age_Max=('age',max),Age_Min=('age',min))
print(grouped_single)
```

18.
```
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','1
5/09/1997'],
    '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))
```

19.
```
import pandas as pd

# World alcohol consumption data
w_a_con = pd.read_csv('world_alcohol.csv')
print("World alcohol consumption sample data:")
print(w_a_con.head())
print('\nShape of the dataframe: ',w_a_con.shape)
print('\nNumber of rows: ',w_a_con.shape[0])
print('\nNumber of column: ',w_a_con.shape[1])
print("\nExtract Column Names:")
print(w_a_con.columns)
```

20.
```
import pandas as pd

df = pd.DataFrame({
    'name_code': ['c0001','1000c','b00c2', 'b2c02', '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("\nIndex of a substring in a specified column of a
dataframe:")
df['Index'] = list(map(lambda x: x.find('c', 0, 5),
df['name_code']))
print(df)
```

21.
```
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)
```

22.
```
import matplotlib.pyplot as plt

X = range(1, 50)
Y = [value * 3 for value in X]
print("Values of X:")
print(*range(1,50))
print("Values of Y (thrice of X):")
print(Y)
# Plot lines and/or markers to the Axes.
plt.plot(X, Y)
# Set the x axis label of the current axis.
plt.xlabel('x - axis')
# Set the y axis label of the current axis.
plt.ylabel('y - axis')
# Set a title
plt.title('Draw a line.')
# Display the figure.
plt.show()
```

23.
```
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3]
# y axis values
y = [2,4,1]
# Plot lines and/or markers to the Axes.
plt.plot(x, y)
# Set the x axis label of the current axis.
plt.xlabel('x - axis')
# Set the y axis label of the current axis.
plt.ylabel('y - axis')
# Set a title
plt.title('Sample graph!')
# Display a figure.
plt.show()
```

24.
```
import matplotlib.pyplot as plt

import pandas as pd
df = pd.read_csv('fdata.csv', sep=',', parse_dates=True,
index_col=0)
df.plot()
plt.show()
```

25.
```
import matplotlib.pyplot as plt

# line 1 points
```

```
x1 = [10,20,30]
y1 = [20,40,10]
# line 2 points
x2 = [10,20,30]
y2 = [40,10,30]
# Set the x axis label of the current axis.
plt.xlabel('x - axis')
# Set the y axis label of the current axis.
plt.ylabel('y - axis')
# Set a title
plt.title('Two or more lines with different widths and colors with
suitable legends ')
# Display the figure.
plt.plot(x1,y1, color='blue', linewidth = 3,  label = 'line1-width-
3')
plt.plot(x2,y2, color='red', linewidth = 5,  label = 'line2-width-
5')
# show a legend on the plot
plt.legend()
plt.show()
```

26.
```
import matplotlib.pyplot as plt

fig = plt.figure()
fig.subplots_adjust(bottom=0.020, left=0.020, top = 0.900,
right=0.800)

plt.subplot(2, 1, 1)
plt.xticks(()), plt.yticks(())

plt.subplot(2, 3, 4)
plt.xticks(())
plt.yticks(())

plt.subplot(2, 3, 5)
plt.xticks(())
plt.yticks(())

plt.subplot(2, 3, 6)
plt.xticks(())
plt.yticks(())

plt.show()
```

27.
```
import matplotlib.pyplot as plt

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

```
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
x_pos = [i for i, _ in enumerate(x)]
plt.bar(x_pos, popularity, color='blue')
plt.xlabel("Languages")
plt.ylabel("Popularity")
plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct
2017 compared to a year ago")
plt.xticks(x_pos, x)
# Turn on the grid
plt.minorticks_on()
plt.grid(which='major', linestyle='-', linewidth='0.5', color='red')
# Customize the minor grid
plt.grid(which='minor', linestyle=':', linewidth='0.5',
color='black')
plt.show()
```

28.
```
import matplotlib.pyplot as plt

x = ['Java', 'Python', 'PHP', 'JS', 'C#', 'C++']
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
x_pos = [i for i, _ in enumerate(x)]
plt.barh(x_pos, popularity, color='green')
plt.xlabel("Popularity")
plt.ylabel("Languages")
plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct
2017 compared to a year ago")
plt.yticks(x_pos, x)
# Turn on the grid
plt.minorticks_on()
plt.grid(which='major', linestyle='-', linewidth='0.5', color='red')
# Customize the minor grid
plt.grid(which='minor', linestyle=':', linewidth='0.5',
color='black')
plt.show()
```

29.
```
import matplotlib.pyplot as plt

x = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
x_pos = [i for i, _ in enumerate(x)]

plt.bar(x_pos, popularity, color=['red', 'black', 'green', 'blue',
'yellow', 'cyan'])

plt.xlabel("Languages")
plt.ylabel("Popularity")
```

```
plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct
2017 compared to a year ago")
plt.xticks(x_pos, x)
# Turn on the grid
plt.minorticks_on()
plt.grid(which='major', linestyle='-', linewidth='0.5', color='red')
# Customize the minor grid
plt.grid(which='minor', linestyle=':', linewidth='0.5',
color='black')
plt.show()
```

30.
```
import numpy as np
import matplotlib.pyplot as plt

# data to plot
n_groups = 5
men_means = (22, 30, 33, 30, 26)
women_means = (25, 32, 30, 35, 29)

# create plot
fig, ax = plt.subplots()
index = np.arange(n_groups)
bar_width = 0.35
opacity = 0.8

rects1 = plt.bar(index, men_means, bar_width,
alpha=opacity,
color='g',
label='Men')

rects2 = plt.bar(index + bar_width, women_means, bar_width,
alpha=opacity,
color='r',
label='Women')

plt.xlabel('Person')
plt.ylabel('Scores')
plt.title('Scores by person')
plt.xticks(index + bar_width, ('G1', 'G2', 'G3', 'G4', 'G5'))
plt.legend()

plt.tight_layout()
plt.show()
```

31.
```
import numpy as np
```

```
import matplotlib.pyplot as plt

N = 5
menMeans = (22, 30, 35, 35, 26)
womenMeans = (25, 32, 30, 35, 29)
menStd = (4, 3, 4, 1, 5)
womenStd = (3, 5, 2, 3, 3)
# the x locations for the groups
ind = np.arange(N)
# the width of the bars
width = 0.35

p1 = plt.bar(ind, menMeans, width, yerr=menStd, color='red')
p2 = plt.bar(ind, womenMeans, width,
bottom=menMeans, yerr=womenStd, color='green')

plt.ylabel('Scores')
plt.xlabel('Groups')
plt.title('Scores by group\n' + 'and gender')
plt.xticks(ind, ('Group1', 'Group2', 'Group3', 'Group4', 'Group5'))
plt.yticks(np.arange(0, 81, 10))
plt.legend((p1[0], p2[0]), ('Men', 'Women'))

plt.show()
```

32.
```
import numpy as np
import matplotlib.pyplot as plt

N = 5
menMeans = (22, 30, 35, 35, 26)
womenMeans = (25, 32, 30, 35, 29)
menStd = (4, 3, 4, 1, 5)
womenStd = (3, 5, 2, 3, 3)
# the x locations for the groups
ind = np.arange(N)
# the width of the bars
width = 0.35

p1 = plt.bar(ind, menMeans, width, yerr=menStd, color='red')
p2 = plt.bar(ind, womenMeans, width,
bottom=menMeans, yerr=womenStd, color='green')

plt.ylabel('Scores')
plt.xlabel('Groups')
plt.title('Scores by group\n' + 'and gender')
plt.xticks(ind, ('Group1', 'Group2', 'Group3', 'Group4', 'Group5'))
plt.yticks(np.arange(0, 81, 10))
```

```
plt.legend((p1[0], p2[0]), ('Men', 'Women'))

plt.show()
```

33.
```
import matplotlib.pyplot as plt

import numpy as np
x = np.random.randn(50)
y = np.random.randn(50)
plt.scatter(x, y, s=70, facecolors='none', edgecolors='g')
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```

34.
```
import math

import random
import matplotlib.pyplot as plt
# create random data
no_of_balls = 25
x = [random.triangular() for i in range(no_of_balls)]
y = [random.gauss(0.5, 0.25) for i in range(no_of_balls)]
colors = [random.randint(1, 4) for i in range(no_of_balls)]
areas = [math.pi * random.randint(5, 15)**2 for i in
range(no_of_balls)]
# draw the plot
plt.figure()
plt.scatter(x, y, s=areas, c=colors, alpha=0.85)
plt.axis([0.0, 1.0, 0.0, 1.0])
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```

35.
```
import matplotlib.pyplot as plt

import pandas as pd
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, label='Math marks', color='r')
plt.scatter(marks_range, science_marks, label='Science marks',
color='g')
plt.title('Scatter Plot')
plt.xlabel('Marks Range')
plt.ylabel('Marks Scored')
```

```
plt.legend()
plt.show()
```

36.
```
import matplotlib.pyplot as plt
import numpy as np
weight1=[67,57.2,59.6,59.64,55.8,61.2,60.45,61,56.23,56]
height1=[101.7,197.6,98.3,125.1,113.7,157.7,136,148.9,125.3,114.9]
weight2=[61.9,64,62.1,64.2,62.3,65.4,62.4,61.4,62.5,63.6]
height2=[152.8,155.3,135.1,125.2,151.3,135,182.2,195.9,165.1,125.1]
weight3=[68.2,67.2,68.4,68.7,71,71.3,70.8,70,71.1,71.7]
height3=[165.8,170.9,192.8,135.4,161.4,136.1,167.1,235.1,181.1,177.3
]
weight=np.concatenate((weight1,weight2,weight3))
height=np.concatenate((height1,height2,height3))
plt.scatter(weight, height, marker='*', color=['blue'])
plt.xlabel('weight', fontsize=16)
plt.ylabel('height', fontsize=16)
plt.title('Group wise Weight vs Height scatter plot',fontsize=20)
plt.show()
```

37.
```
import pandas as pd
df = pd.DataFrame({'X':[78,85,96,80,86],
'Y':[84,94,89,83,86],'Z':[86,97,96,72,83]});
print(df)
```

38.
```
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)
```

39.
```
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("First three rows of the data frame:")
print(df.iloc[:3])
```

```
40. 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("Select specific columns:")
print(df[['name', 'score']])
```