

# Experiment 1

## Aim- working with formulas and function

### Basic formulas

#### 1. SUM

The SUM() formula performs addition on selected cells. It works on cells containing numerical values and requires two or more cells.

```
=SUM(C2:C5)
```

#### MIN and MAX

The MIN() formula requires a range of cells, and it returns the minimum value.

```
=MIN(E2:E5)
```

#### AVERAGE

The AVERAGE() formula calculates the average of selected cells.

```
=AVERAGE(C2:C5)
```

#### COUNT

The COUNT() formula counts the total number of selected cells. It will not count the blank cells and different data formats other than numeric.

```
=COUNT(E2:E5)
```

#### IF

The IF Excel formula is straightforward. It is similar to an if-else statement in a programming language. We will provide the logic of the formula. If the logic is correct, it will return a certain value; if the logic is False, it will return a different value.

```
=IF(G2<24.9,"Fit","Unfit")
```

The screenshot shows a Microsoft Excel worksheet titled 'New Microsoft Excel Worksheet (2) - Excel'. The worksheet contains a table with student data and calculated statistics. The table has columns for name, marks1, marks2, sum, MIN, MAX, AVERAGE, COUNT, and IF. The data is as follows:

name	marks1	marks2	sum	MIN	MAX	AVERAGE	COUNT	IF
ss	50	88	138	50	88	69	2	PASS
ff	45	45	90	45	45	45	2	PASS
ee	32	44	76	32	44	38	2	FAIL
gg	11	16	27	11	16	13.5	2	FAIL

The IF column uses the formula `=IF(G2<24.9,"Fit","Unfit")` to determine if the student is 'Fit' or 'Unfit' based on their average mark. The status is 'PASS' for students with an average mark of 69 or higher, and 'FAIL' for students with an average mark of 45 or lower.

Result- we were able to perform basic operations in

# Experiment2

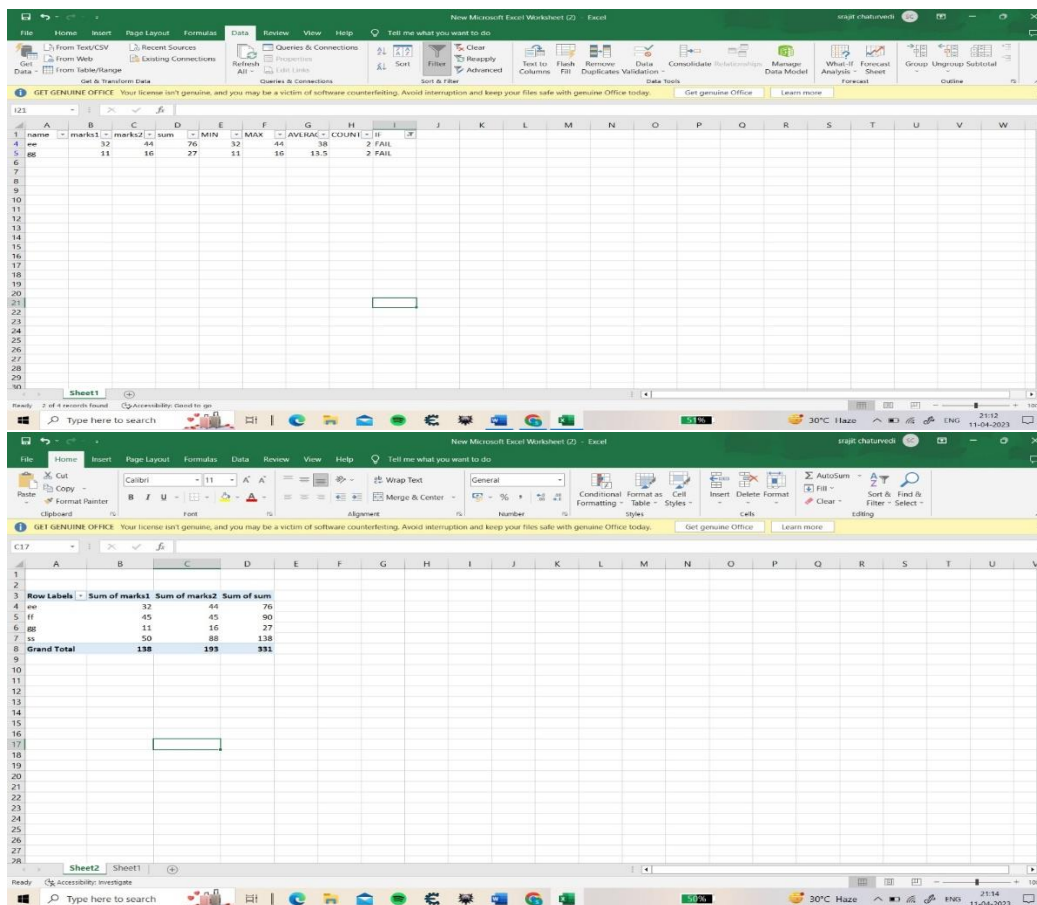
**Aim** – working with filter and pivot table

## Filter

1. Click a cell in the range or table that you want to filter.
2. On the Data tab, click Filter.
3. Click the arrow. in the column that contains the content that you want to filter.
4. Under Filter, click Choose One, and then enter your filter criteria.

## Pivot table

1. Select the cells you want to create a PivotTable from. ...
2. Select Insert > PivotTable.
3. This will create a PivotTable based on an existing table or range. ...
4. Choose where you want the PivotTable report to be placed. ...
5. Click OK.



**Result**- we are able to perform filter and pivot table

## Experiment 3

**Aim** – charts using excel (bar chart ,pie chart ,scatter plot)

### **Barchart**

#### **Steps-**

Select the range

On the Insert tab, in the Charts group, click the Column symbol.

Click Clustered Bar.

### **Pie chart**

#### **Steps to –**

1. Click Insert > Chart. ...
2. Click Pie and then double-click the pie chart you want.
3. In the spreadsheet that appears, replace the placeholder data with your own information. ...
4. When you've finished, close the spreadsheet.
5. Click the chart and then click the icons next to the chart to add finishing touches:

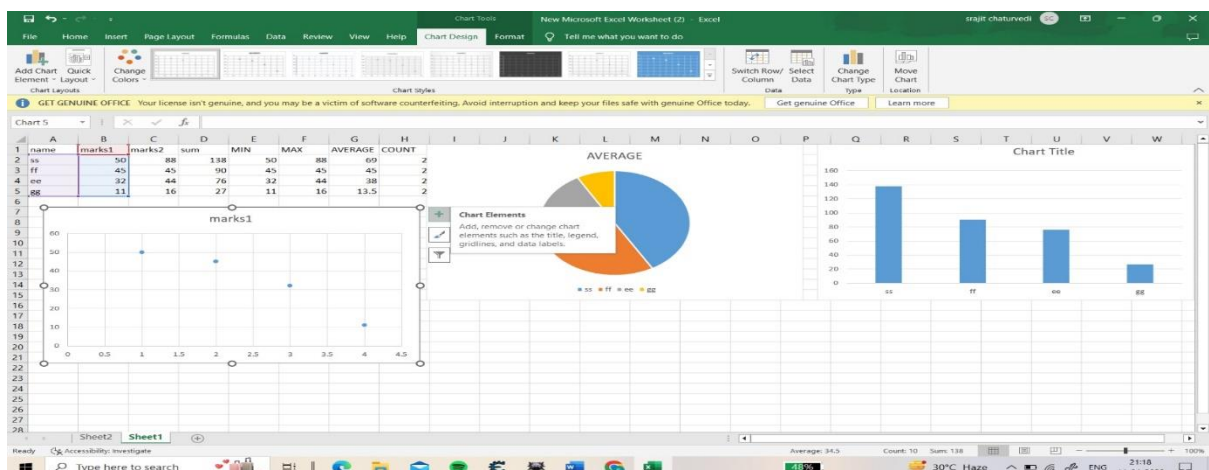
### **Scatter plot**

#### **Steps-**

**Choose your independent and dependent variables**

Click the Insert tab, and then click Insert Scatter (X, Y) or Bubble Chart.

Click Scatter.



**Result-** we were able to perform the charts successfully

## Experiment 4

**Aim-** demonstrate the histogram and descriptive statistics

### Histogram

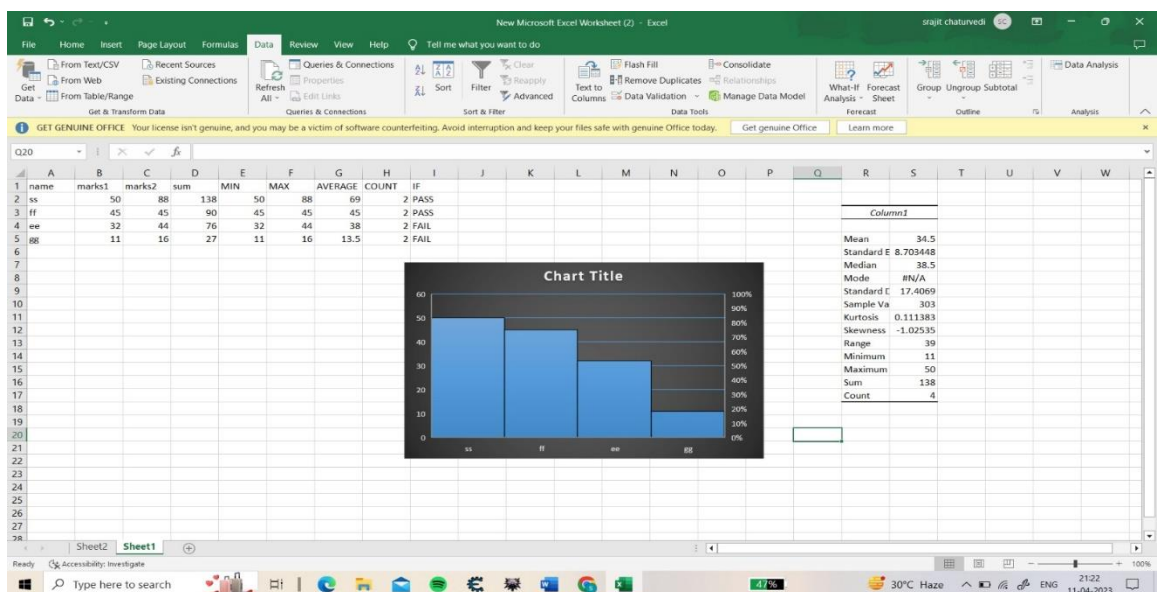
**Steps-**

1. Make sure you [load the Analysis ToolPak](#) to add the **Data Analysis** command to the **Data** tab.
2. On a worksheet, type the input data in one column, and the bin numbers in ascending order in another column.
3. Click **Data > Data Analysis > Histogram > OK**.
4. Under **Input**, select the input range (your data), then select the bin range.
5. Under **Output options**, choose an output location.
6. To show the data in descending order of frequency, click **Pareto (sorted histogram)**.
7. To show cumulative percentages and add a cumulative percentage line, click **Cumulative Percentage**.
8. To show an embedded histogram chart, click **Chart Output**.

### Descriptive Statistics

**Steps-**

1. On the Data tab, in the Analysis group, click Data Analysis. Note: can't find the Data Analysis button? ...
2. Select Descriptive Statistics and click OK.
3. Select the range A2:A15 as the Input Range.
4. Select cell C1 as the Output Range.
5. Make sure Summary statistics is checked.
6. Click OK.



**Result-** we were able to perform the the operation sucessfully

# Experiment 5

**Aim-** exponential smoothing and moving average

## Moving average

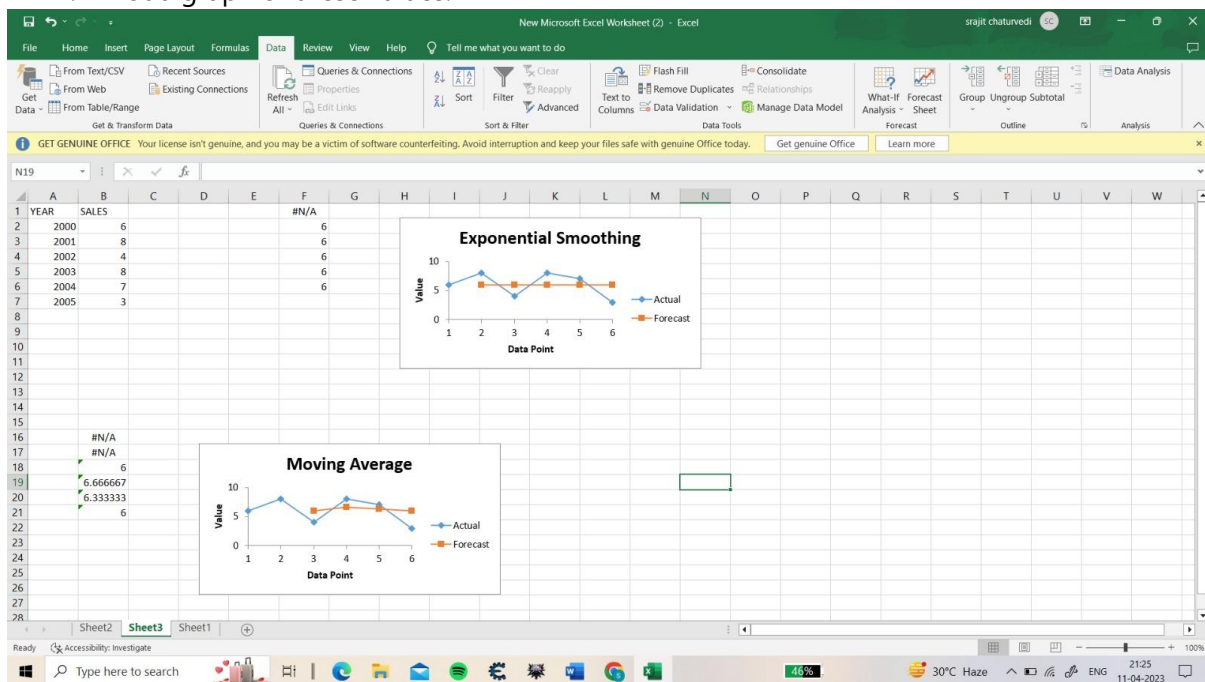
### **Steps**

1. Create a time series in Excel. A time series is a data point series arranged according to a time order. ...
2. Select "Data Analysis" ...
3. Choose "Moving Average" ...
4. Select your interval, input and output ranges. ...
5. Create a graph using the values.

## Exponential smoothing

### **Steps**

1. We must first click on the "Data" tab and "Data Analysis."
2. After that, select the "Exponential Smoothing" option
3. Click in the Input Range box and select the range
4. Click in the Damping factor box
5. Click in the Output Range box
6. Click OK.
7. Plot a graph of these values.



**Result**-we were able to perform the the operation successfully

# Experiment 6

**Aim**-introduction to numpy and implementing array using numpy in python

## Numpy-

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

## Implementing numpy

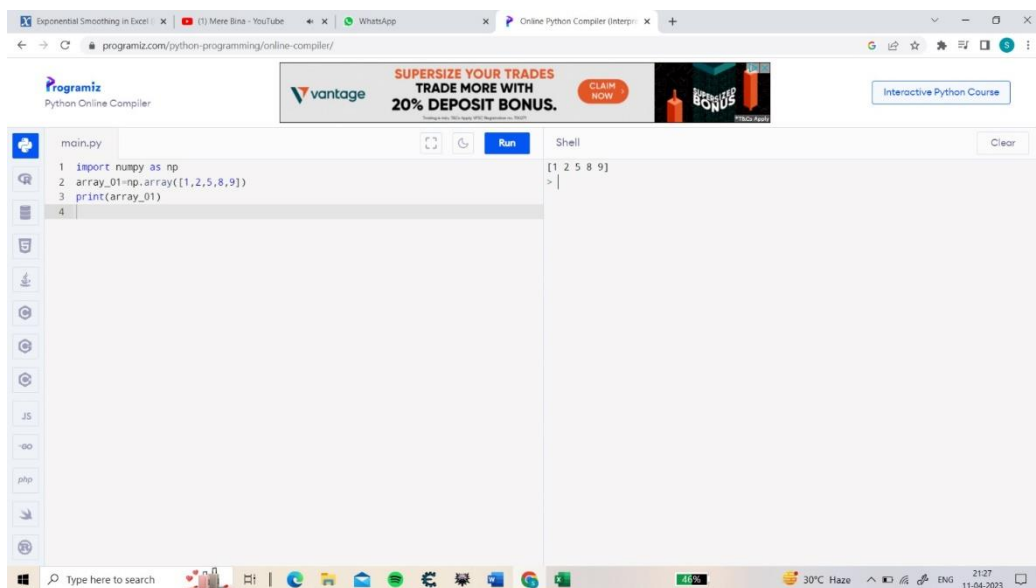
### 1. Array creating Code:

```
import numpy as np
```

```
array_01=np.array([1,2,5,8,9])
```

```
print(array_01)
```

**output:**



### Reshape Code:

```
import numpy as np
```

```
array_01=np.array([1,2,5,8,9,10,12,5,3])
```

```
print(array_01)
```

```
new_array=array_01.reshape(3,3)
```

```
print()
```

```
print(new_array)
```

## 2. Change data type Code:

```
import numpy as np
```

```
x=np.array([1,2,3,4,45,67,89,80])
```

```
print("data type:",x.dtype)
```

```
print(x)
```

```
x1=np.array([1,2,3,4,45,67,89,80],dtype='f')
```

```
print(x1.dtype)
```

```
print(x1)
```

output:

The screenshot shows the Programiz Python Online Compiler interface. The code editor on the left contains the following Python code:

```
main.py
1 import numpy as np
2 x=np.array([1,2,3,4,45,67,89,80])
3 print("data type:",x.dtype)
4 print(x)
5 x1=np.array([1,2,3,4,45,67,89,80],dtype='f')
6 print(x1.dtype)
7 print(x1)
8
```

The output shell on the right displays the results of the code execution:

```
data type: int64
[ 1  2  3  4 45 67 89 80]
float32
[ 1.  2.  3.  4. 45. 67. 89. 80.]
>
```

The browser's address bar shows the URL `programiz.com/python-programming/online-compiler/`. The top navigation bar includes the Programiz logo, a sponsored banner for SBI Net Bank, and a button for "Interactive Python Course". The bottom status bar shows the system clock as 21:29 on 11-04-2023.

### 3. Arrange function Code:

import numpy as np

arr=np.arange(10,100,10)

print(arr)

**output:**

The screenshot shows the Programiz Python Online Compiler interface. The code editor on the left contains the following Python code:

```
main.py
1 import numpy as np
2 arr=np.arange(10,100,10)
3 print(arr)
4
```

The output shell on the right displays the results of the code execution:

```
[1.  1.2 1.4 1.6 1.8 2.  2.2 2.4 2.6 2.8 3.  3.2 3.4 3.6 3.8 4.  4.2 4.4
 4.6 4.8 5.  5.2 5.4 5.6 5.8 6.  6.2 6.4 6.6 6.8 7.  7.2 7.4 7.6 7.8 8.
 8.2 8.4 8.6 8.8 9.  9.2 9.4 9.6 9.8]>
```

The browser's address bar shows the URL `programiz.com/python-programming/online-compiler/`. The top navigation bar includes the Programiz logo, a sponsored banner for Vantage, and a button for "Interactive Python Course". The bottom status bar shows the system clock as 21:30 on 11-04-2023.



#### 4. Arithmetics Code:

```
import numpy as np

x2=np.array([1,2,3,4])

x3=np.array([1,2,3,4])

varadd=np.add(x2,x3)

print("Addition:",varadd)

varsub=np.subtract(x2,x3)

print("subtractio:",varsub)

varmulti=np.multiply(x2,x3)

print("multiplication:",varmulti)

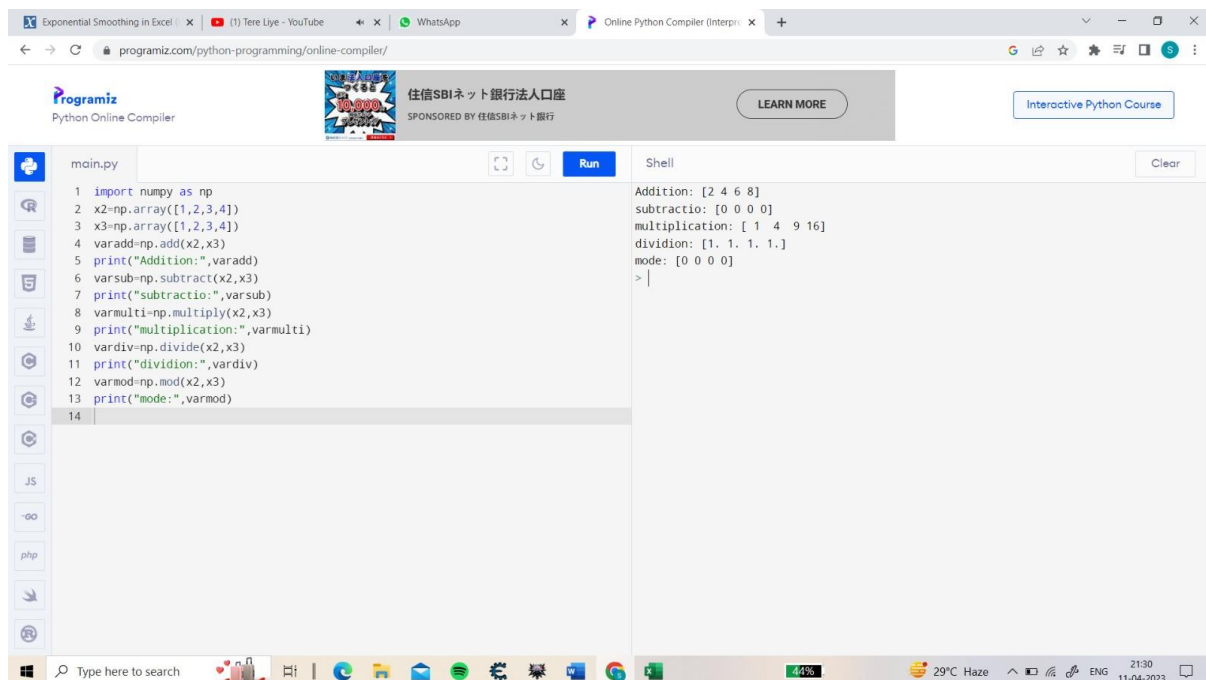
vardiv=np.divide(x2,x3)

print("dividion:",vardiv)

varmod=np.mod(x2,x3)

print("mode:",varmod)
```

#### output:



The screenshot shows a web browser window with the URL `programiz.com/python-programming/online-compiler/`. The page features the Programiz logo and a navigation bar. The main content area displays a Python script in a code editor, which has been executed. The output of the script is shown in a shell window on the right side of the editor. The script performs various arithmetic operations on two arrays, `x2` and `x3`, both containing the values `[1, 2, 3, 4]`. The operations include addition, subtraction, multiplication, division, and modulo. The output of the script is as follows:

```
Addition: [2 4 6 8]
subtractio: [0 0 0 0]
multiplication: [1 4 9 16]
dividion: [1. 1. 1. 1.]
mode: [0 0 0 0]
```

**Result**-we were able to perform the the numpy operation sucessfully

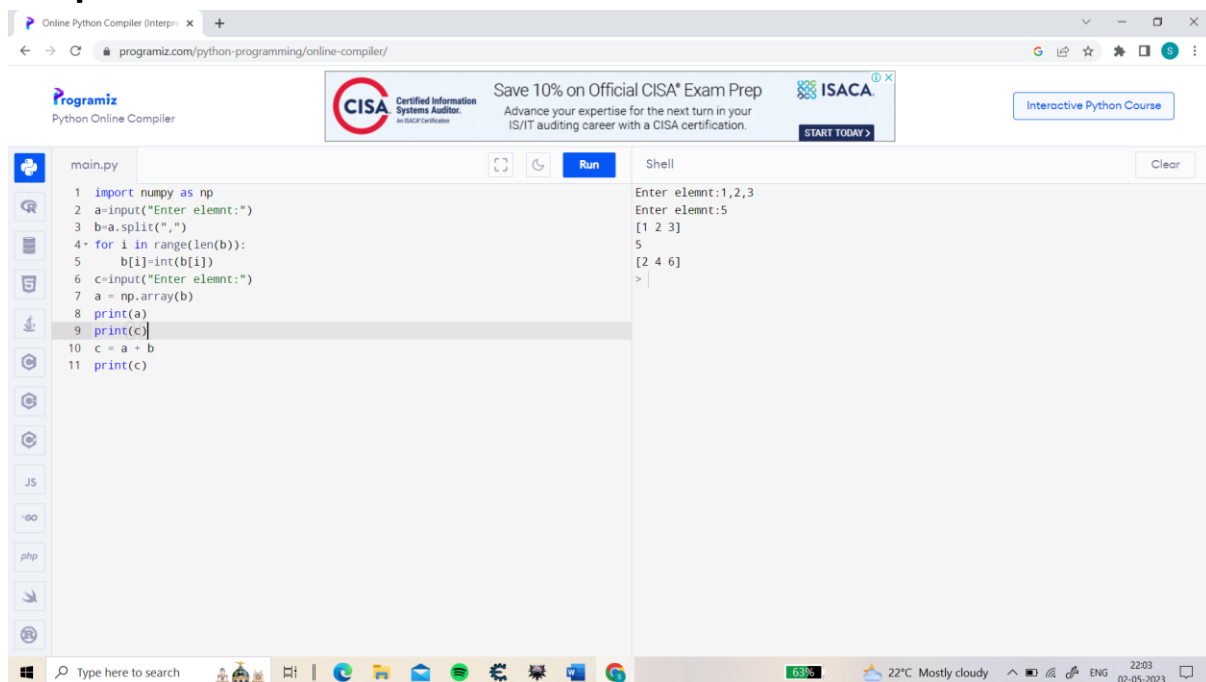
## Experiment 8

**Aim-** Implement Numpy Broadcasting

**Code-**

```
import numpy as np
a=input("Enter elemnt:")
b=a.split(",")
for i in range(len(b)):
    b[i]=int(b[i])
c=input("Enter elemnt:")
a = np.array(b)
print(a)
print(c)
c = a + b
print(c)
```

**Output-**



The screenshot shows a web browser window with the URL `programiz.com/python-programming/online-compiler/`. The page features a header with logos for CISA, ISACA, and Programiz, along with promotional text for CISA exam prep and a link to an interactive Python course. The main area is divided into a code editor on the left and a shell on the right. The code editor contains the following Python code:

```
1 import numpy as np
2 a=input("Enter elemnt:")
3 b=a.split(",")
4 for i in range(len(b)):
5     b[i]=int(b[i])
6 c=input("Enter elemnt:")
7 a = np.array(b)
8 print(a)
9 print(c)
10 c = a + b
11 print(c)
```

The shell on the right shows the execution output:

```
Enter elemnt:1,2,3
Enter elemnt:5
[1 2 3]
5
[2 4 6]
>
```

The bottom of the image shows a Windows taskbar with a search bar, several application icons, and system status information including 63% battery, 22°C temperature, and the date 02-05-2023.

**Result=** we were able to perform broadcasting in numpy array

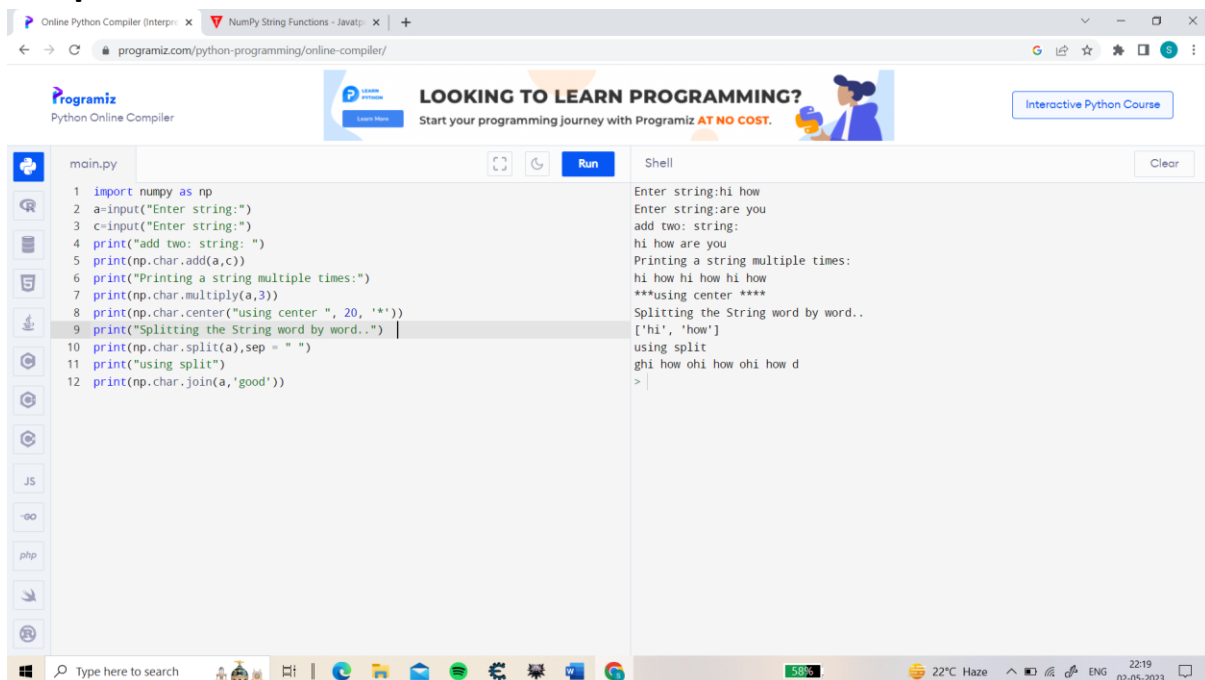
## Experiment-9

**Aim-** Implement numpy strings function add, multiply, center, split, join

**Code-**

```
import numpy as np
a=input("Enter string:")
c=input("Enter string:")
print("add two: string: ")
print(np.char.add(a,c))
print("Printing a string multiple times:")
print(np.char.multiply(a,3))
print(np.char.center("using center ", 20, '*'))
print("Splitting the String word by word..")
print(np.char.split(a),sep = " ")
print("using split")
print(np.char.join(a,'good'))
```

**Output-**

The screenshot shows a web browser window with the URL 'programiz.com/python-programming/online-compiler/'. The page features a 'Programiz Python Online Compiler' interface. On the left, there's a file explorer with 'main.py' selected. The main area displays the Python code from the previous block. On the right, the 'Shell' tab shows the output of the code execution. The output matches the expected results: 'Enter string:hi how', 'Enter string:are you', 'add two: string:', 'hi how are you', 'Printing a string multiple times:', 'hi how hi how hi how', '\*\*\*using center \*\*\*', 'Splitting the String word by word..', ['hi', 'how'], 'using split', and 'ghi how ohi how ohi how d'. The bottom of the browser shows a Windows taskbar with the date '02-05-2023' and time '22:19'.

**Result-** We were able to perform string functions: add,multiply,center,split,join successfully

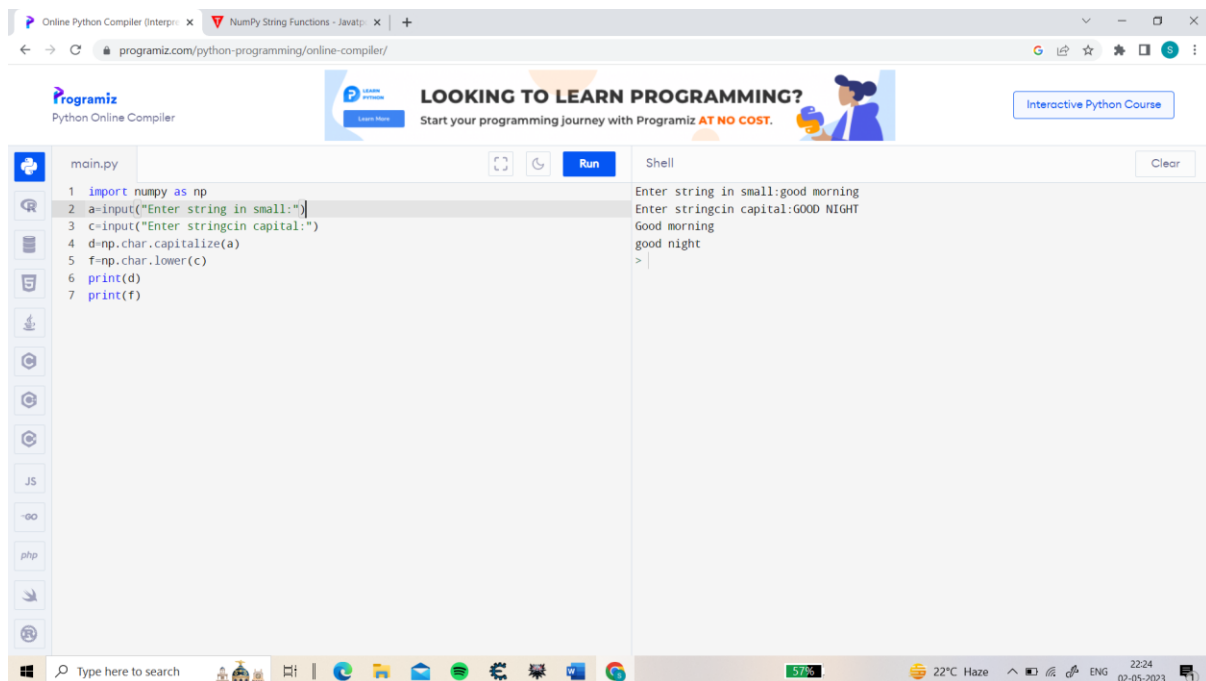
## Experiment -10

**Aim-**Implement Numpy string functions: capitalize and lower function

**Code:**

```
import numpy as np
a=input("Enter string in small:")
c=input("Enter string in capital:")
d=np.char.capitalize(a)
f=np.char.lower(c)
print(d)
print(f)
```

**OUTPUT-**



The screenshot shows the Programiz Online Python Compiler interface. The code editor on the left contains the following Python code:

```
1 import numpy as np
2 a=input("Enter string in small:")
3 c=input("Enter string in capital:")
4 d=np.char.capitalize(a)
5 f=np.char.lower(c)
6 print(d)
7 print(f)
```

The output window on the right shows the results of the program execution:

```
Enter string in small:good morning
Enter string in capital:GOOD NIGHT
Good morning
good night
>
```

The interface also includes a top navigation bar with the Programiz logo, a search bar, and a button to start a programming journey. The bottom status bar shows the system clock as 22:24 on 02-05-2023.

**Result-**We successfully implemented Numpy string functions: capitalize and lower

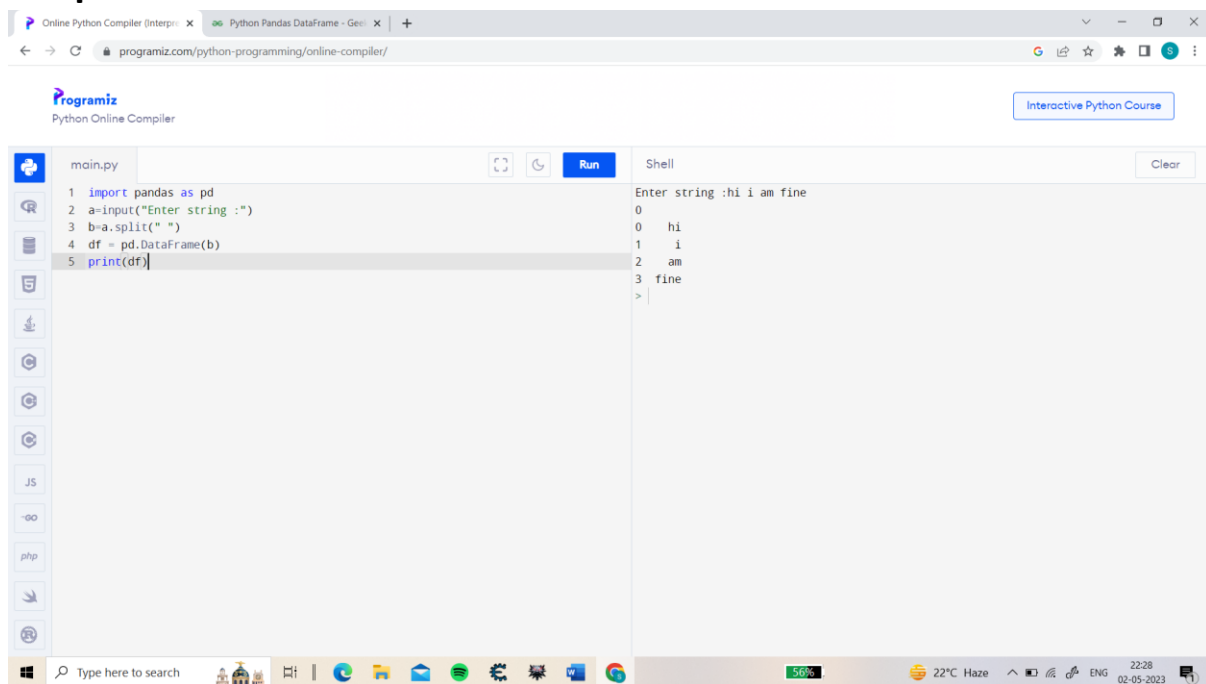
## Experiment 11

**Aim-**Implement data frames using PANDAS library

**Code-**

```
import pandas as pd
a=input("Enter string :")
b=a.split(" ")
df = pd.DataFrame(b)
print(df)
```

**Output**



The screenshot shows a web browser window with the URL `programiz.com/python-programming/online-compiler/`. The page title is "Programiz Python Online Compiler". There is a button labeled "Interactive Python Course". The main area is divided into two panels. The left panel, titled "main.py", contains the following Python code:

```
1 import pandas as pd
2 a=input("Enter string :")
3 b=a.split(" ")
4 df = pd.DataFrame(b)
5 print(df)
```

The right panel, titled "Shell", shows the output of the code execution:

```
Enter string :hi i am fine
0    hi
1     i
2    am
3   fine
>
```

The bottom of the image shows a Windows taskbar with various icons and system information: "22°C Haze", "22:28", and "02-05-2023".

**Result-** We successfully implemented Data Frames using PANDAS Library

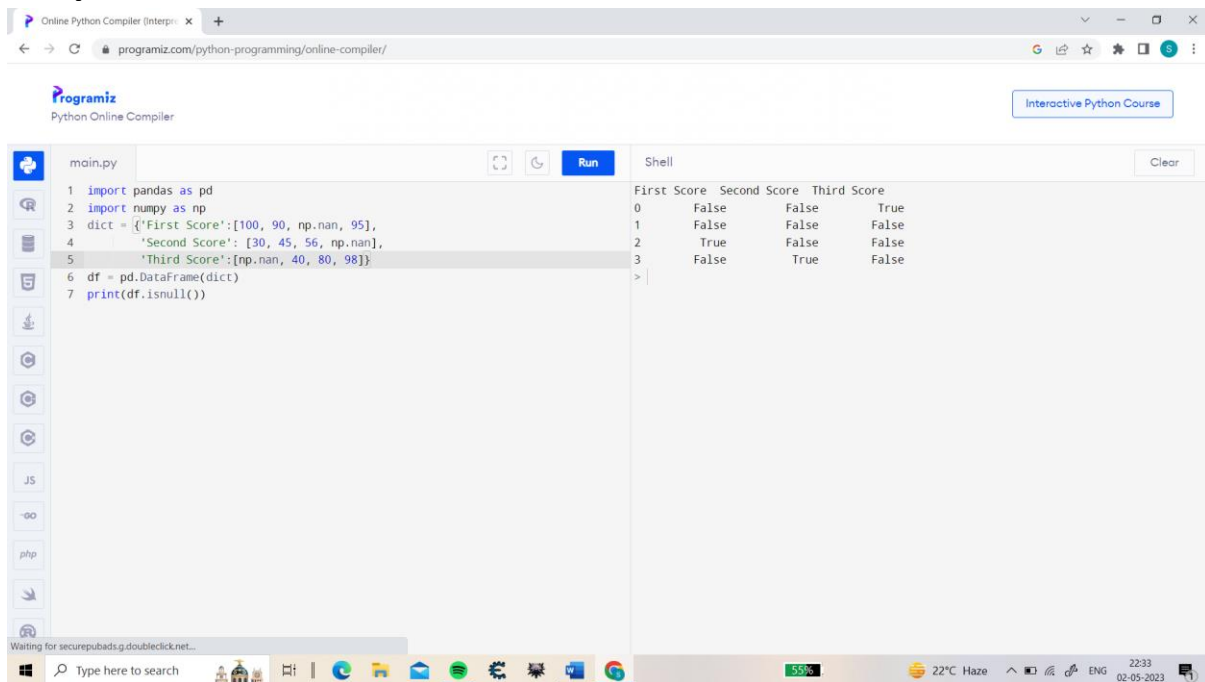
## Experiment -12

**Aim-** Implement pandas library for working with missing values

**Code-**

```
import pandas as pd
import numpy as np
dict = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}
df = pd.DataFrame(dict)
print(df.isnull())
```

**Output-**



The screenshot shows the Programiz Python Online Compiler interface. The code editor on the left contains the following Python code:

```
1 import pandas as pd
2 import numpy as np
3 dict = {'First Score':[100, 90, np.nan, 95],
4         'Second Score': [30, 45, 56, np.nan],
5         'Third Score':[np.nan, 40, 80, 98]}
6 df = pd.DataFrame(dict)
7 print(df.isnull())
```

The output shell on the right displays the result of the code execution:

	First Score	Second Score	Third Score
0	False	False	True
1	False	False	False
2	True	False	False
3	False	True	False

The interface also includes a file explorer on the left with icons for Python, JS, and PHP, and a status bar at the bottom showing system information like temperature and time.

**Result-** We successfully implemented Pandas library for working with missing values.

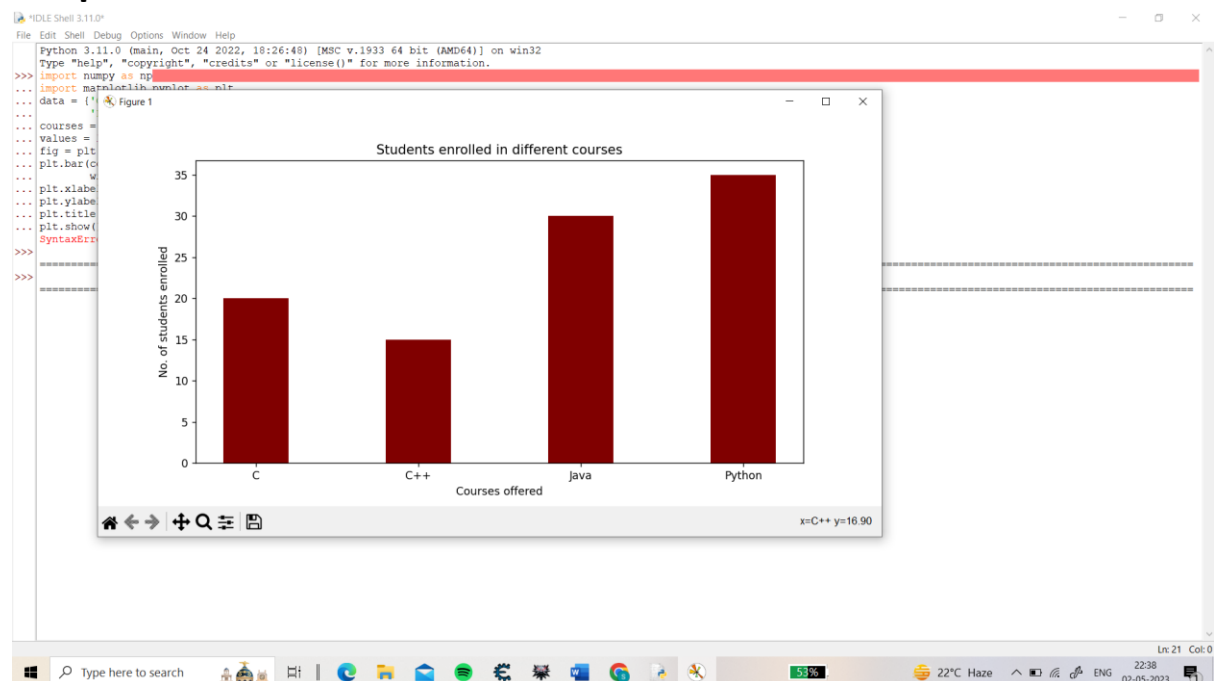
## Experiment-13

**Aim-**Implement bar chart using Matplotlib library for data visualization.

### Code-

```
import numpy as np
import matplotlib.pyplot as plt
data = {'C':20, 'C++':15, 'Java':30,
        'Python':35}
courses = list(data.keys())
values = list(data.values())
fig = plt.figure(figsize = (10, 5))
plt.bar(courses, values, color='maroon',
        width = 0.4)
plt.xlabel("Courses offered")
plt.ylabel("No. of students enrolled")
plt.title("Students enrolled in different courses")
plt.show()
```

### Output-



**Result**-We successfully implemented Bar Chart using Matplotlib library

## Experiment 14

**Aim**-Implement scatter plot using matplotlib library for data visualization.

**Code**-

```
import matplotlib.pyplot as plt
```

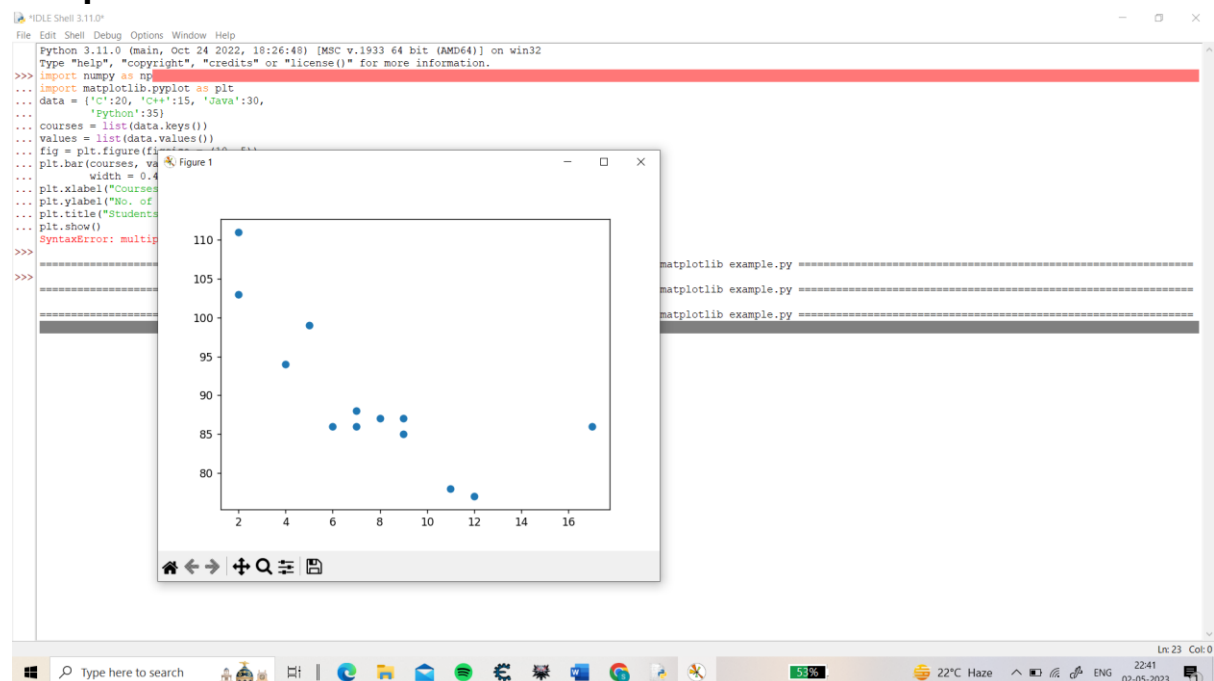
```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
```

```
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
```

```
plt.scatter(x, y)
```

```
plt.show()
```

**Output**-



**Result**- We successfully implemented scatter plot using matplotlib library for data visualization.



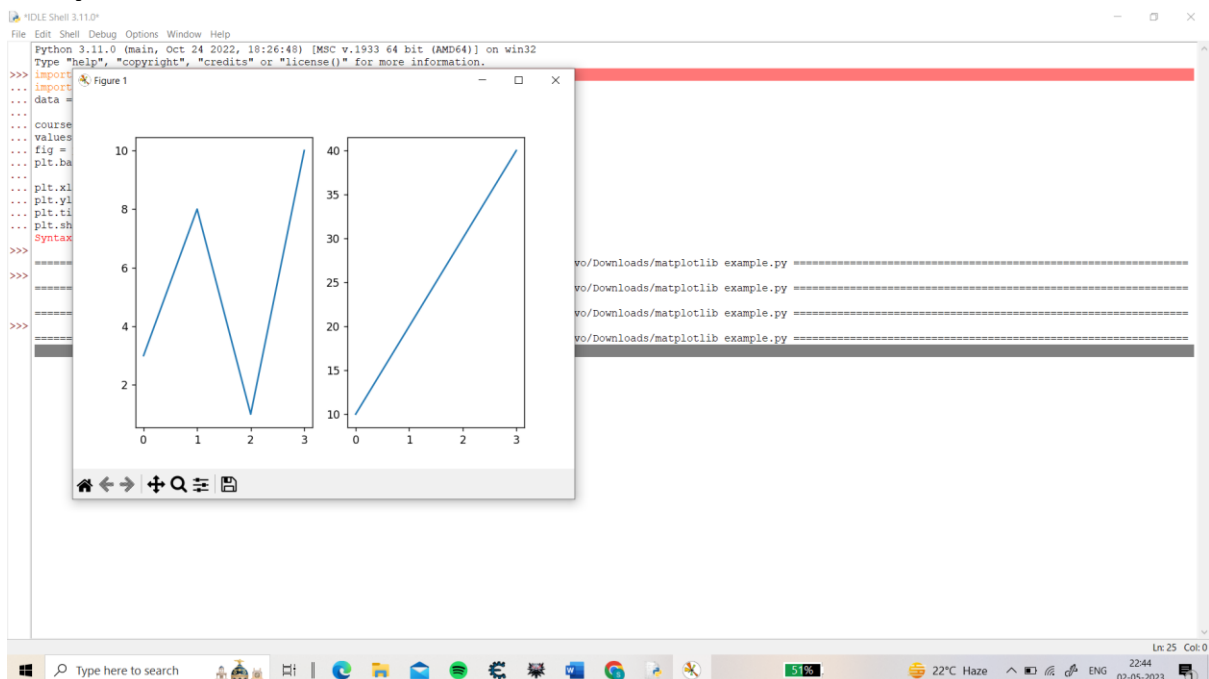
## Experiment -15

**Aim-**Implement sub plot using matplotlib library for data visualization

**Code-**

```
import matplotlib.pyplot as plt
import numpy as np
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])
plt.subplot(1, 2, 1)
plt.plot(x,y)
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])
plt.subplot(1, 2, 2)
plt.plot(x,y)
plt.show()
```

**Output-**



**Result-** We successfully Implemented sub plot using matplotlib library for data visualization.