BANGALORE INSTITUTE OF TECHNOLOGY

K.R.ROAD, V.V.PURAM, BANGALORE-560 004

Department of Computer Science & Engineering

Data Visualization using Python
Laboratory Manual

III SEM

BCS358D

Prepared By:

Dr. Girija J & Dr. Madhuri J

Dept. of CS&E

Bangalore Institute of Technology

K. R. Road, V.V. Pura, Bengaluru 560004

Department of Computer Science and Engineering

COURSE LEARNING OBJECTIVES (CLO)

This laboratory course enables students to get practical experience in design, develop, implement, analyze and evaluation/testing of

CLO 1. Demonstrate the use of IDLE or PyCharm IDE to create Python Applications

CLO 2. Using Python programming language to develop programs for solving real-world problems

CLO 3. Implementation of Matplotlib for drawing different Plots
CLO 4. Demonstrate working with Seaborn, Bokeh.

CLO 5. Working with Plotly for 3D, Time Series and Maps.

COURSE OUTCOMES (CO)

On the completion of this laboratory course, the students will be able to:

CO 1. Demonstrate the use of IDLE or PyCharm IDE to create Python Applications
CO 2. Use Python programming constructs to develop programs for solving real-world problems
CO 3. Use Matplotlib for drawing different Plots
CO 4. Demonstrate working with Seaborn, Bokeh for visualization**.**
CO 5. Use Plotly for drawing Time Series and Maps.

		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
BCS358D	CO1		3			3							
	CO2	3	3	3		3							
	CO3					3							
	CO4					3							
	CO5					3							

		PSO1	PSO2
BCS358D	CO1	2	
	CO2	2	2
	CO3	2	
	CO4	2	

	CO5	2	

DATA VISUALIZATION WITH PYTHON

Subject Code: BCS358D CIE Marks: 50

Hours/Week: 0:0:2 Exam Hours: 03

List of problems for which student should develop program and execute in the Laboratory:

Sl. No.	Name of Experiment (Part-A)
1.	a) Write a python program to find the best of two test average marks out of three test's marks accepted from the user. b) Develop a Python program to check whether a given number is palindrome or not and also count the number of occurrences of each digit in the input number.
2.	a) Defined as a function F as Fn = Fn-1 + Fn-2. Write a Python program which accepts a value for N (where N >0) as input and pass this value to the function. Display suitable error message if the condition for input value is not followed. b) Develop a python program to convert binary to decimal, octal to hexadecimal using functions.
3.	a) Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters and lowercase letters. b)Write a Python program to find the string similarity between two given strings Sample Output: Original string: Original string: Python Exercises Python Exercises Python Exercises Python Exercise Similarity between two said strings: 1.0 Similarity between two said strings: 0.96774193548
4.	a) Write a Python program to Demonstrate how to Draw a Bar Plot using Matplotlib. b) Write a Python program to Demonstrate how to Draw a Scatter Plot using Matplotlib.
5.	a) Write a Python program to Demonstrate how to Draw a Histogram Plot using Matplotlib. b) Write a Python program to Demonstrate how to Draw a Pie Chart using Matplotlib.
6.	a) Write a Python program to illustrate Linear Plotting using Matplotlib. b) Write a Python program to illustrate liner plotting with line formatting using Matplotlib.
7.	Write a Python program which explains uses of customizing seaborn plots with Aesthetic functions.
8.	Write a Python program to explain working with bokeh line graph using Annotations and Legends. a) Write a Python program for plotting different types of plots using Bokeh.
9.	Write a Python program to draw 3D Plots using Plotly Libraries.

10.	a) Write a Python program to draw Time Series using Plotly Libraries. b) Write a Python program for creating Maps using Plotly Libraries.

PROGRAM -1

1a) Write a Python program to find the best of two test average marks out of three test marks accepted by the user.

```
m1 = int(input("Enter marks for test1 : "))
m2 = int(input("Enter marks for test2 : "))
m3 = int(input("Enter marks for test3 : "))
        if m1 <= m2 and m1 <= m3:
            avgMarks = (m2+m3)/2
        elif m2 <= m1 and m2 <= m3:
            avgMarks = (m1+m3)/2
        elif m3 <= m1 and m2 <= m2:
            avgMarks = (m1+m2)/2
print("Average of best two test marks  out of three test's marks is", avgMarks);
```

OUTPUT

```
Enter marks for test1: 45
Enter marks for test2: 39
Enter marks for test3: 48
Average of best two test marks out of three test's marks is 46.5
```

1b) Develop a Python program to check whether a given number is palindrome or not and also count the number of occurrences of each digit in the input number.

```
val = int(input("Enter a value : "))
str_val = str(val)
if str_val == str_val[::-1]:
   print("Palindrome")
else:
   print("Not Palindrome")
for i in range(10):
        if str_val.count(str(i)) > 0:
                print(str(i),"appears", str_val.count(str(i)), "times");
```

OUTPUT:

```
Enter a value: 1234234
        Not Palindrome
        1 appears 1 times
        2 appears 2 times
        3 appears 2 times
        4 appears 2 times
Enter a value: 12321
        Palindrome
        1 appears 2 times
        2 appears 2 times
        3 appears 1 times
```

PROGRAM -2

2a) Defined as a function F as Fn = Fn-1 + Fn-2. Write a Python program that accepts a value for N (where N >0) as input and pass this value to the function. Display a suitable error message if the condition for input value is not followed.

```
def fn(n):
    if n == 1:
        return 0
    elif n == 2:
        return 1
    else:
        return fn(n-1) + fn(n-2)

num = int(input("Enter a number : "))

if num > 0:
    print("fn(", num, ") = ",fn(num) , sep ="")
else:
    print("Error in input")
```

OUTPUT:

Enter a number : 5

fn(5) = 3

Enter a number : -5

Error in input

2b) Develop a python program to convert binary to decimal, octal to hexadecimal using functions.

```
def bin_to_dec():
    bin=int(input("Enter a binary number : "))
    dec=0
    i=0
    while bin!=0:
        dec+=(bin%10)*(2**i)
        i+=1
        bin//=10
    print("Decimal Equivalent is : ",dec)


def oct_to_hex():
    oct=int(input("Enter an octal number : "))
    dec=0
    i=0
    while oct!=0:
        dec+=(oct%10)*(8**i)
        i+=1
        oct//=10
    hex =""
    while dec!=0:
        rem=dec%16
        if rem<10:
            hex+=str(rem)
        else:
            hex+= chr(ord('A')+rem-10)
        dec//=16
    hex=hex[::-1]
    print("Hexadecimal Equivalent is : ", hex)


print("BINARY TO DECIMAL CONVERSION")
bin_to_dec()
print("\n\nOCTAL TO HEXADECIMAL CONVERSION")
oct_to_hex()
```

PROGRAM -3

3a) Write a Python program that accepts a sentence and find the number of words, digits, uppercase letters, and lowercase letters.

```
sentence = input("Enter a sentence: ")
words = digits = upper = lower = 0

# Splitting the sentence using split() method , by default split is by spaces
#   Return value - list of strings

split_sentence = sentence.split()
print("The result of split() on input sentence is : \n"+str(split_sentence)+"\n")

words =  len(split_sentence )

for c in sentence:
   if c.isdigit():
      digits = digits + 1
   elif c.isupper():
      upper = upper + 1
   elif c.islower():
      lower = lower + 1

print ("No of Words: ", words)
print ("No of Digits: ", digits)
print ("No of Uppercase letters: ", upper)
print ("No of Lowercase letters: ", lower)
```

OUTPUT:

Enter a sentence: Rama went to Devaraja market to pick 2 kgs of vegetable

The result of split() on input sentence is :

['Rama', 'went', 'to', 'Devaraja', 'market', 'to', 'pick', '2', 'kgs', 'of', 'vegetable']

No of Words: 11

No of Digits: 1

No of Uppercase letters: 2

No of Lowercase letters: 42

3b) Write a Python program to find the string similarity between two given strings.

```
str1 = input("Enter String 1 \n")
str2 = input("Enter String 2 \n")

if len(str2) < len(str1):
    short = len(str2)
    long = len(str1)
else:
    short = len(str1)
    long = len(str2)
matchCnt = 0
for i in range(short):
    if str1[i] == str2[i]:
        matchCnt += 1
print("Similarity between two said strings:")
print(matchCnt/long)
```

OUTPUT:

```
Enter String 1
Python Exercises
Enter String 2
Python Exercises
Similarity between two said strings:
1.0
Enter String 1
Python Exercises
Enter String 2
Python Exercise
Similarity between two said strings: 0.9375
```

Matplotlib

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations in a variety of formats. It is widely used for producing high-quality plots and charts in scientific computing, data analysis, and machine learning. Matplotlib provides a range of functions for creating different types of plots, including line plots, scatter plots, bar plots, histograms, and more. Different visualizations plots are as follows:

Scatter plots: Scatter plots are particularly useful when exploring the relationship between two continuous variables. They excel at revealing patterns, trends, and correlations between data points. These visualizations are adept at identifying outliers, showcasing them as points deviating from the main cluster. By providing a clear picture of the distribution of data points along two axes, scatter plots aid in understanding the spread and density of values. Moreover, they are valuable for comparing different datasets, recognizing similarities or differences.

Histogram: A histogram is a graphical representation of the distribution of a dataset, typically used for continuous or discrete data. It provides a way to visualize the frequency or count of data points within specific intervals or bins. In a histogram, the data is divided into contiguous, non-overlapping intervals, and the height of each bar in the chart represents the frequency or count of data points falling within that interval.

To create a histogram, you divide the range of the data into bins or intervals and then count the number of data points that fall into each bin. The resulting bar chart, with the bars representing these counts, provides a visual summary of the data's distribution.

Bar chart: A bar chart is a graphical representation of data in which rectangular bars are used to represent the values of different categories. Each bar's length is proportional to the value it represents. Bar charts are effective for comparing discrete categories or groups and are particularly useful for showing the distribution of categorical data.

Pie chart: Pie charts are a type of data visualization that is commonly used to represent the proportions of different parts of a whole. The primary purpose of a pie chart is to show the relationship of parts to a whole and to illustrate how each part contributes to the total.

Seaborn

Seaborn is a statistical data visualization library built on top of Matplotlib in Python. It provides an interface for creating informative and attractive statistical graphics. Seaborn comes with several built-in themes and color palettes to make it easy to create aesthetically pleasing visualizations. It is particularly useful for exploring complex datasets and understanding relationships between variables.

Bokeh

Bokeh is a Python interactive visualization library that targets modern web browsers for presentation. It allows you to create interactive, web-ready visualizations in Python. Bokeh

generates HTML and JavaScript code that can be embedded into web pages. This allows you to create interactive visualizations that can be easily shared on the web.

Plotly

Plotly is a versatile Python library for creating interactive and publication-quality plots and dashboards. It supports a wide range of chart types. Plotly excels at creating interactive plots. Users can zoom, pan, hover over data points for additional information, and perform other interactive actions directly within the plot. Its ability to create web-based dashboards makes it a powerful tool for building data-driven applications.

Dataset used

1. Cars.csv

Id	Model	Price	Age	Mfg_Month	Mfg_Year	KM	Fuel_Type	HP	Met_Color	Automatic	cc	Doors
1	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13500	23	10	2002	46986	Diesel	90	1	0	2000	3
2	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13750	23	10	2002	72937	Diesel	90	1	0	2000	3
3	?TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	13950	24	9	2002	41711	Diesel	90	1	0	2000	3
4	TOYOTA Corolla 2.0 D4D HATCHB TERRA 2/3-Doors	14950	26	7	2002	48000	Diesel	90	0	0	2000	3
5	TOYOTA Corolla 2.0 D4D HATCHB SOL 2/3-Doors	13750	30	3	2002	38500	Diesel	90	0	0	2000	3

2. Cars_BarPlot

Car	Sales
Audi	419
BMW	263
Mercedes	330
Honda	760

3. tips.csv

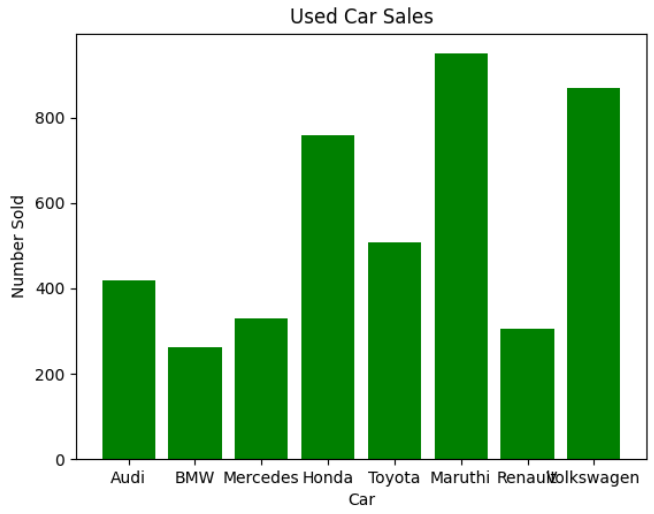
	total_bill	tip	sex	smoker	day	time	size
1	16.99	1.01	Female	No	Sun	Dinner	2
2	10.34	1.66	Male	No	Sun	Dinner	3
3	21.01	3.5	Male	No	Sun	Dinner	3
4	23.68	3.31	Male	No	Sun	Dinner	2
5	24.59	3.61	Female	No	Sun	Dinner	4

4. Rainfall_data

Year	Month	Day	Specific Humidity	Relative Humidity	Temperature	Precipitation
2000	1	1	8.06	48.25	23.93	0
2000	2	1	8.73	50.81	25.83	0.11
2000	3	1	8.48	42.88	26.68	0.01
2000	4	1	13.79	55.69	22.49	0.02
2000	5	1	17.4	70.88	19.07	271.14

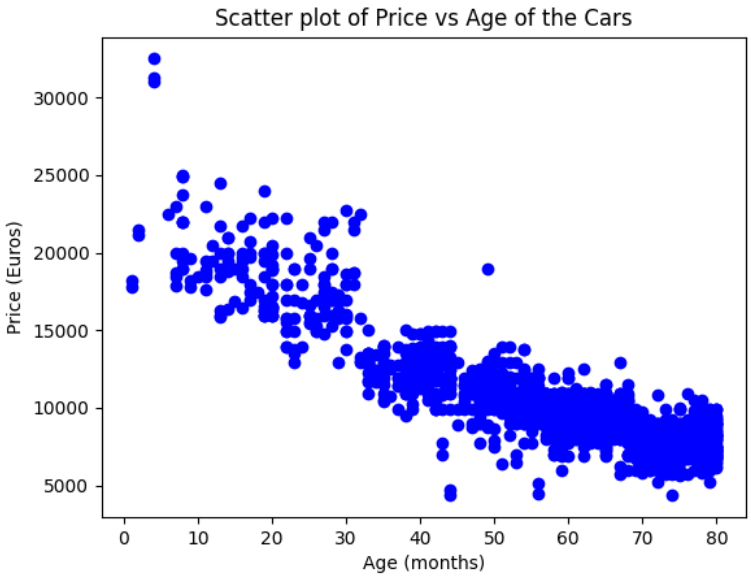
4a) Write a Python program to Demonstrate how to Draw a Bar Plot using Matplotlib

```
# Import the necessary modules
import matplotlib.pyplot as plt
import pandas as pd
# Initialize the lists for X and Y
data = pd.read_csv("Car_Barplot.csv")
df = pd.DataFrame(data)
X = list(df.iloc[:, 0])
Y = list(df.iloc[:, 1])

# Plot the data using bar() method
plt.bar(X, Y, color='g')
plt.title("Used Car Sales")
plt.xlabel("Car")
plt.ylabel("Number Sold")
# Show the plot
plt.show()
```



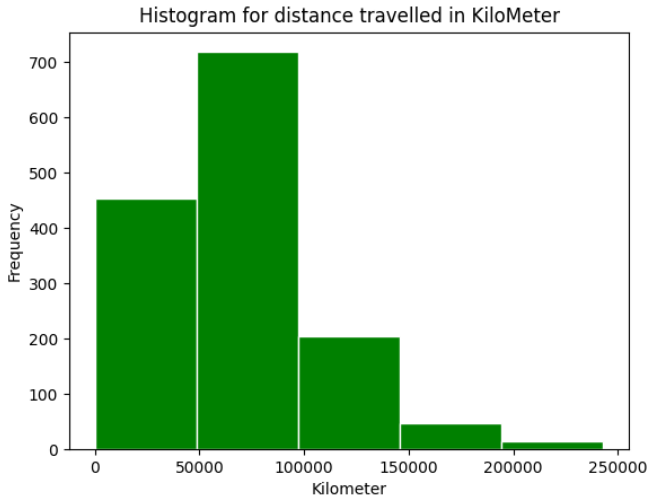
4. b) Write a Python program to Demonstrate how to draw a Scatter Plot using Matplotlib

```
# import the necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


# Importing data.
cars_data = pd.read_csv("Toyota.csv")

# Create scatter plot using two variables, Age and Price.
plt.scatter(cars_data['Age'],cars_data['Price'],c='blue')
# To set the title
plt.title('Scatter plot of Price vs Age of the Cars')
# To set the x and y axis labels.
plt.xlabel('Age (months)')
plt.ylabel('Price (Euros)')
# To show the scatter plot
plt.show()
```



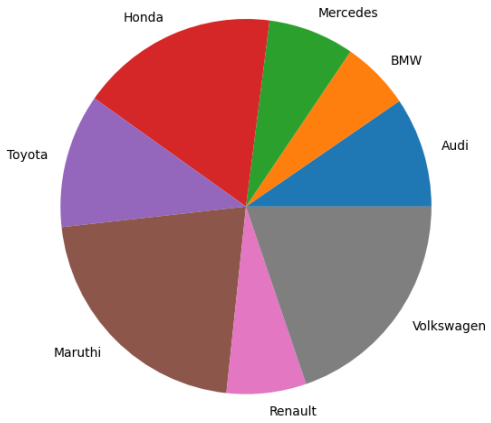
5. a) Write a Python program to Demonstrate how to draw a Histogram using Matplotlib

```
# import the necessary libraries
# Pandas library for data frames
import pandas as pd

# numpy library to do numerical operations

import numpy as np
import matplotlib.pyplot as plt

cars_data = pd.read_csv("cars.csv")
plt.title('Histogram for distance travelled in KiloMeter')

plt.hist(cars_data ['KM'], color='green', edgecolor='white', bins=5)
plt.xlabel('Kilometer')
plt.ylabel('Frequency')
plt.show()
```



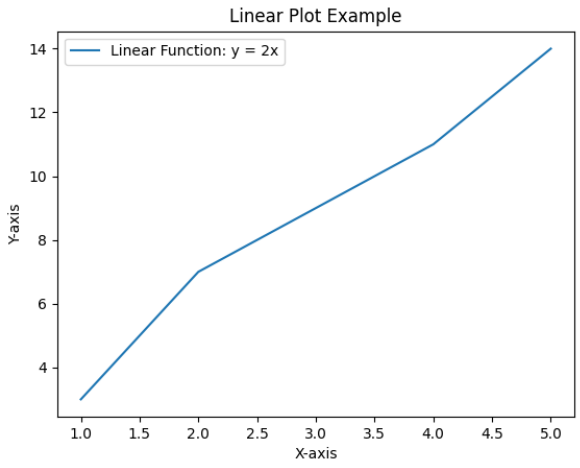
5. b) Write a Python program to Demonstrate how to draw a Piechart using Matplotlib

```
# Import libraries
import matplotlib.pyplot as plt
import pandas as pd

# Creating dataset
cars_data = pd.read_csv("Car_BarPlot.csv")
cars = cars_data["Car"]
data = cars_data["Sales"]

# Creating plot
fig = plt.figure(figsize =(10, 7))
plt.pie(data, labels = cars)

# show plot
plt.show()
```



6a. Write a Python program to illustrate Linear Plotting using Matplotlib

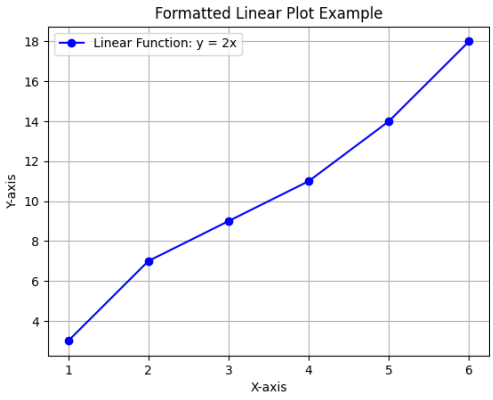
```
import matplotlib.pyplot as plt
def linear_plot():
    # Sample data
    x = [1, 2, 3, 4, 5]
    y = [3, 7, 9, 11, 14]
    # Plotting the data
    plt.plot(x, y, label='Linear Function: y = 2x')

    # Adding labels and title
    plt.xlabel('X-axis')
    plt.ylabel('Y-axis')
    plt.title('Linear Plot Example')
    plt.legend()
    plt.show()

# Call the function to generate the plot
linear_plot()
```



6b) Write a Python program to illustrate liner plotting with line formatting using Matplotlib

```
import matplotlib.pyplot as plt
def formatted_linear_plot():
    # Sample data
    x = [1, 2, 3, 4, 5, 6]
    y = [3, 7, 9, 11, 14, 18]
    plt.plot(x, y, marker='o', linestyle='-', color='b', label='Linear Function: y = 2x')

# Adding labels and title
    plt.xlabel('X-axis')
    plt.ylabel('Y-axis')
    plt.title('Formatted Linear Plot Example')

    plt.legend()
    plt.grid(True)  # Add a grid for better readability
    plt.show()

# Call the function to generate the formatted linear plot
formatted_linear_plot()
```

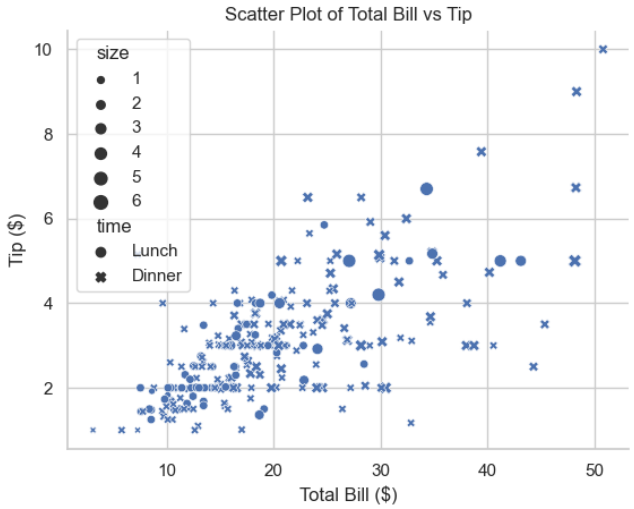
Output



7a) Write a Python program which explains uses of customizing seaborn plots with Aesthetic functions.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Load a sample dataset
tips = sns.load_dataset("tips")

# Set the aesthetic style of the plot
sns.set(style="whitegrid")

# Create a scatter plot using Seaborn
sns.scatterplot(x="total_bill", y="tip",  style="time", size="size", data=tips)

# Customize the plot further using Seaborn aesthetic functions
sns.despine()  # Remove the top and right spines from the plot

# Set custom labels and title
plt.xlabel("Total Bill ($)")
plt.ylabel("Tip ($)")
plt.title("Scatter Plot of Total Bill vs Tip")

# Show the plot
plt.show()
```

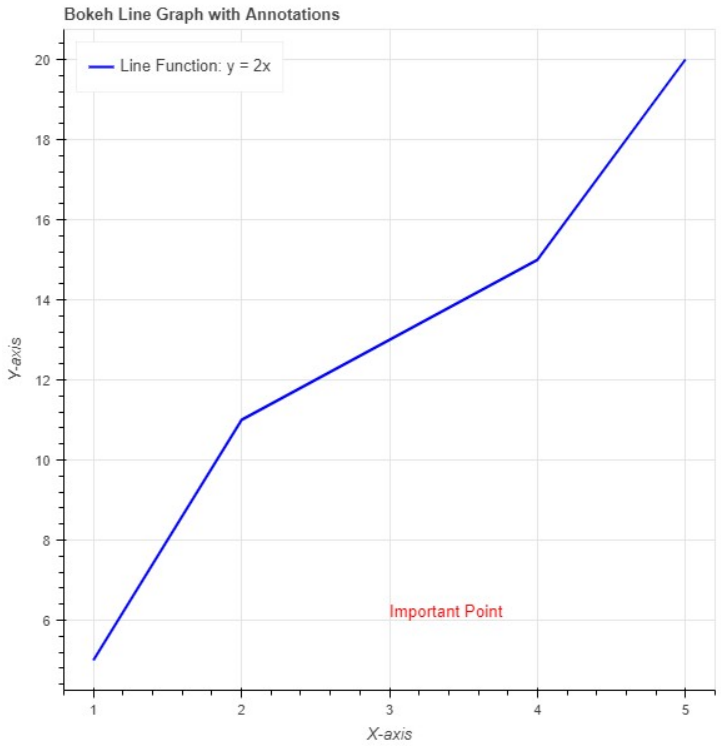
Output


Scatter Plot of Total Bill vs Tip
size
1
2
3
4
5
6
time
Lunch
Dinner
Tip ($)
Total Bill ($)
10
8
6
4
2
10
20
30
40
50

8 a) Write a Python program to explain working with bokeh line graph using Annotations and Legends.

```
from bokeh.plotting import figure, output_file, show
from bokeh.models import Label

# Sample data
x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]

# Output to static HTML file
output_file("line_graph_with_annotations.html")

# Create a figure
p = figure(title="Bokeh Line Graph with Annotations", x_axis_label='X-axis', y_axis_label='Y-axis')

# Plot the line
p.line(x, y, line_width=2, line_color="blue", legend_label="Line Function: y = 2x")

# Add an annotation
annotation = Label(x=3, y=6, text="Important Point", text_font_size="10pt", text_color="red")
p.add_layout(annotation)

# Add legend
p.legend.location = "top_left"
p.legend.click_policy = "hide"

# Show the plot
show(p)
```

Output

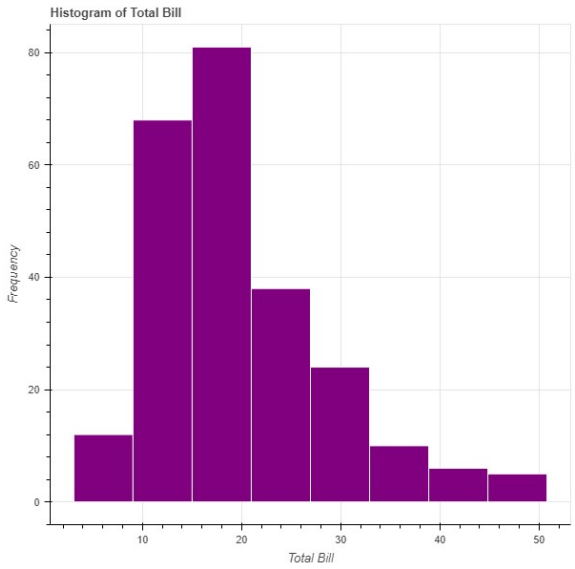

Bokeh Line Graph with Annotations
Line Function: y = 2x
Important Point
Y-axis
X-axis
20
18
16
14
12
10
8
6
1
2
3
4
5

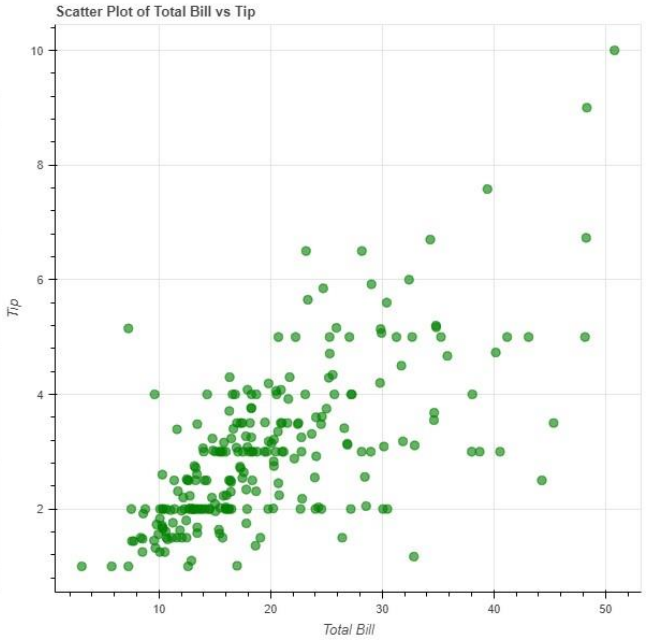
8 b) Write a Python program for plotting different types of plots using Bokeh

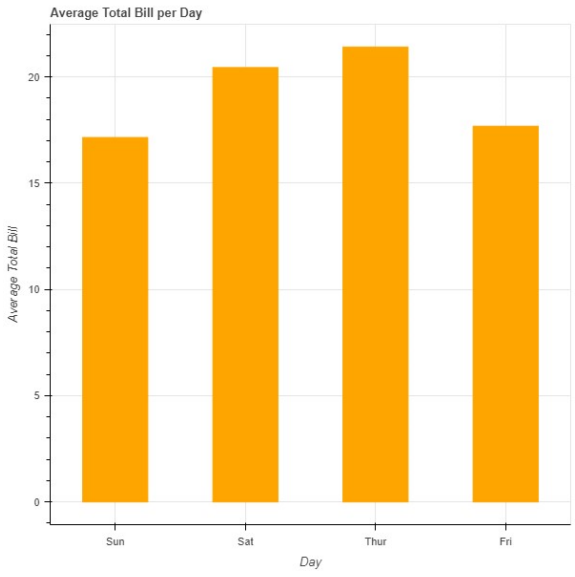
```
import pandas as pd
import numpy as np
from bokeh.plotting import figure, output_file, show
from bokeh.layouts import gridplot

# Load the tips dataset
tips = pd.read_csv("tips.csv")

# Output to static HTML file
output_file("bokeh_tips_plots.html")

# Histogram
hist, edges = np.histogram(tips['total_bill'], bins=8)
hist_plot = figure(title="Histogram of Total Bill", x_axis_label='Total Bill', y_axis_label='Frequency')
hist_plot.quad(top=hist, bottom=0, left=edges[:-1], right=edges[1:], fill_color="purple", line_color="white")

# Bar Plot
day_categories = tips['day'].unique()
average_total_bill = tips.groupby('day')['total_bill'].mean()
bar_plot = figure(title="Average Total Bill per Day", x_axis_label='Day', y_axis_label='Average Total Bill', x_range=day_categories)
bar_plot.vbar(x=day_categories, top=average_total_bill, width=0.5, color="orange")

# Scatter Plot
scatter_plot = figure(title="Scatter Plot of Total Bill vs Tip", x_axis_label='Total Bill', y_axis_label='Tip')
scatter_plot.scatter(x='total_bill', y='tip', size=8, color="green", alpha=0.6, source=tips)
```

```
# Combine plots into a grid
plots = gridplot([[hist_plot, bar_plot], [scatter_plot]])

# Show the combined plot
show(plots)
```

Output



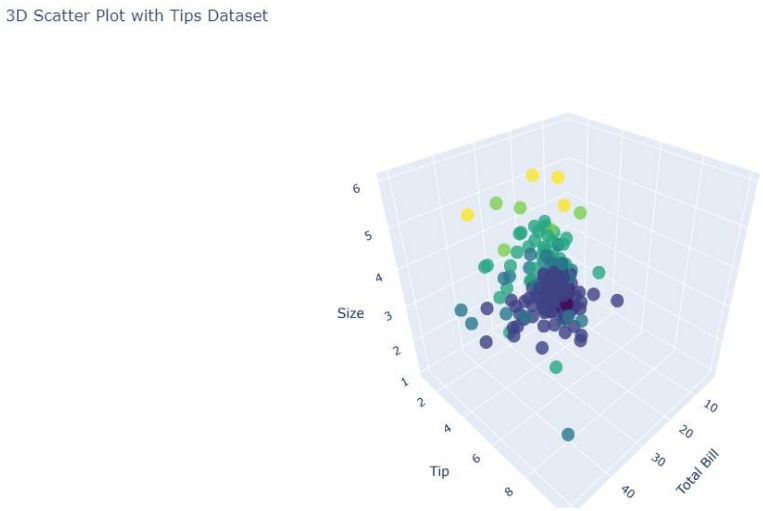




9a) Write a Python program to draw 3D Plots using Plotly Libraries.

```
import plotly.graph_objects as go
import pandas as pd

# Load the tips dataset
tips = pd.read_csv("tips.csv")

# Create a 3D scatter plot
fig = go.Figure()

scatter = go.Scatter3d(
    x=tips['total_bill'],
    y=tips['tip'],
    z=tips['size'],
    mode='markers',
    marker=dict(size=8, color=tips['size'], colorscale='Viridis', opacity=0.8)
)
fig.add_trace(scatter)
# Set axis labels and title
fig.update_layout(scene=dict(xaxis_title='Total Bill', yaxis_title='Tip', zaxis_title='Size'))
fig.update_layout(title='3D Scatter Plot with Tips Dataset')

# Save the plot as an HTML file
fig.write_html("3d_scatter_plot_tips.html")
```

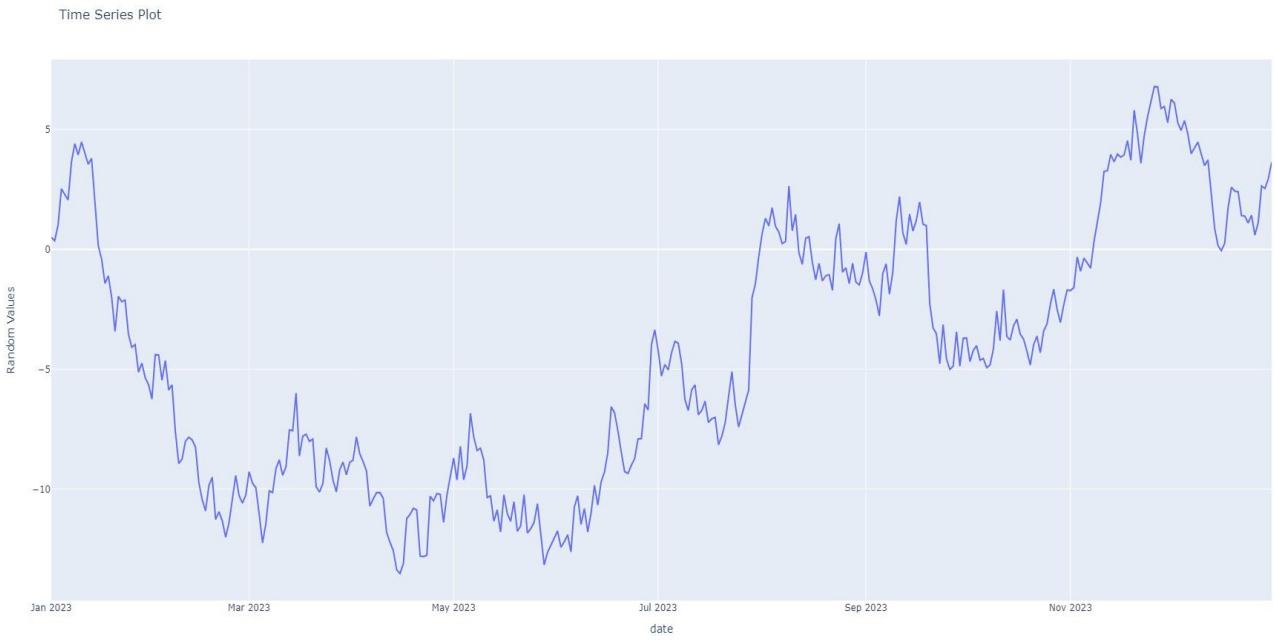
Output



10a). Write a Python program to draw Time Series using Plotly Libraries

```
import plotly.express as px
import pandas as pd

# Load the dataset (replace 'your_dataset.csv' with the actual file path)
data = pd.read_csv("Rainfall_data.csv")

# Combine Year, Month, and Day columns to create a datetime column
data['Date'] = pd.to_datetime(data[['Year', 'Month', 'Day']])

# Create time series plot
fig = px.line(data, x='Date', y=[ 'Temperature'],
title='Time Series Plot', labels={'value': 'Values'}, line_shape='linear')
fig.show()
```

Output



10 b) Write a Python program for creating Maps using Plotly Libraries.

```
import plotly.express as px
# Sample data for demonstration
data = {
    'City': ['New York', 'San Francisco', 'Los Angeles', 'Chicago', 'Houston'],
    'Lat': [40.7128, 37.7749, 34.0522, 41.8781, 29.7604],
    'Lon': [-74.0060, -122.4194, -118.2437, -87.6298, -95.3698],
    'Population': [8175133, 870887, 3971883, 2716000, 2328000]
}
# Create a map
fig = px.scatter_geo(data, lat='Lat', lon='Lon', text='City', size='Population',
                     projection='natural earth', title='Population of Cities')
fig.update_traces(textposition='top center')
fig.show()
```

Output