DVP LAB MANUAL

Q1(A):write a python program to find the best of two tests average marks out of three tests marks accepted from the user

Soln:

```
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 test 1: 45
enter marks for test 2: 39
enter marks for test 3: 48
average of best two test marks out of three tests marks is: 46.5
```

Q1(B):Develop a python program to check whether a given number is palindrome or not and also count the number of occurrence of each digit in the input number.

Soln:

```
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:1001
palindrome
0 appears 2 times
1 appears 2 times
```

Q2(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.

Soln:

```
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 : -1
Error in input
```

Q2(B). Develop a python program to convert binary to decimal,octal to hexadecimal using functions.

Soln:

```
def binary_to_decimal(binary_num):
        decimal_num = int(binary_num, 2)
        return decimal_num
```

```
def octal_to_hexadecimal(octal_num):
        decimal_num = int(octal_num, 8)
        hexadecimal_num = hex(decimal_num).replace('0x', '')
        return hexadecimal_num.upper()

binary_num = input("Enter a binary number: ")
decimal_num = binary_to_decimal(binary_num)
print("Decimal equivalent:", decimal_num)

octal_num = input("Enter an octal number: ")
hexadecimal_num = octal_to_hexadecimal(octal_num)
print("Hexadecimal equivalent:", hexadecimal_num)
```

Output:

```
Enter a binary number : 10111001
185
Enter a octal number : 675
1BD
```

Q3(A):Demonstrate a python program that accepts the sentences and finds the number of words,digits upper case and lower case letter.

Soln:

```
sentence = input("Enter a sentence : ")
wordList = sentence.split(" ")
print("This sentence has", len(wordList), "words")
digCnt = upCnt = loCnt = 0
for ch in sentence:
if '0' <= ch <= '9':
digCnt += 1
elif 'A' <= ch <= 'Z':
upCnt += 1
elif 'a' <= ch <= 'z':
loCnt += 1
print("This sentence has", digCnt, "digits", upCnt, "upper case letters", loCnt, "lower case letters")
```

Output:

```
Enter a sentence : Welcome 2 VtuCode
This sentence has 3 words
```

This sentence has 0 digits 1 upper case letters 0 lower case letters
This sentence has 0 digits 1 upper case letters 1 lower case letters
This sentence has 0 digits 1 upper case letters 2 lower case letters
This sentence has 0 digits 1 upper case letters 3 lower case letters
This sentence has 0 digits 1 upper case letters 4 lower case letters
This sentence has 0 digits 1 upper case letters 5 lower case letters
This sentence has 0 digits 1 upper case letters 6 lower case letters
This sentence has 0 digits 1 upper case letters 6 lower case letters
This sentence has 1 digits 1 upper case letters 6 lower case letters
This sentence has 1 digits 1 upper case letters 6 lower case letters
This sentence has 1 digits 2 upper case letters 6 lower case letters
This sentence has 1 digits 2 upper case letters 7 lower case letters
This sentence has 1 digits 2 upper case letters 8 lower case letters
This sentence has 1 digits 3 upper case letters 8 lower case letters
This sentence has 1 digits 3 upper case letters 9 lower case letters
This sentence has 1 digits 3 upper case letters 10 lower case letters
This sentence has 1 digits 3 upper case letters 11 lower case letters

Q3(B):).Write a python program to find the string similarity between two given strings.

Soln:

```
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:
Welcome to vtucode
```

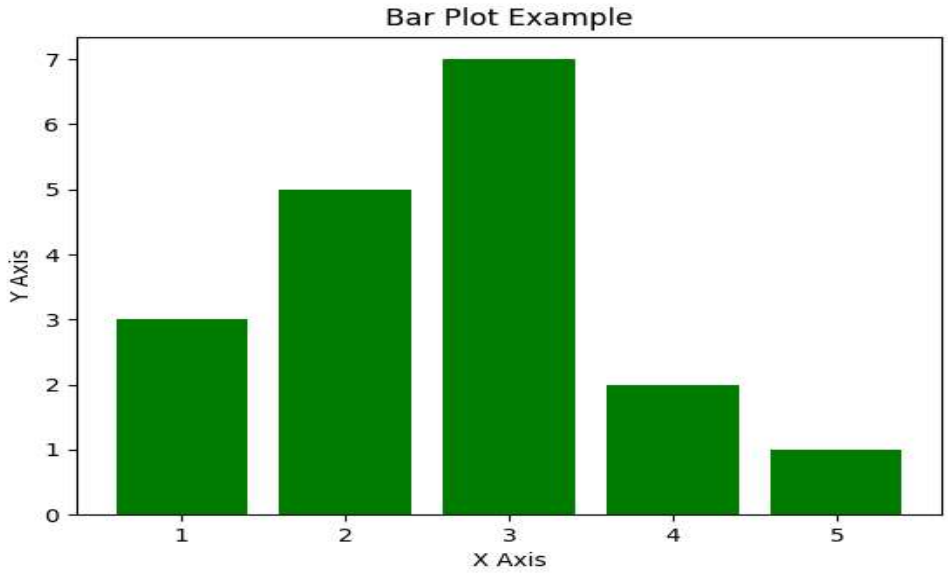
Enter String 2:
Welcome to vtucode
Similarity between two said strings:
1.0
Enter String 1:
Welcome to vtucode
Enter String 2:
author vtucode
Similarity between two said strings:
0.1111111111111111

Q4(A):write a python program to demonstrate how to draw a bar plot using mathplotlib.

Soln:

```
import matplotlib.pyplot as pltx = [1, 2, 3, 4, 5]
y = [3, 5, 7, 2, 1]
plt.bar(x, y, color='green')
plt.title('Bar Plot Example')
plt.xlabel('X Axis')
plt.ylabel('Y Axis')
plt.show()
```

Output:

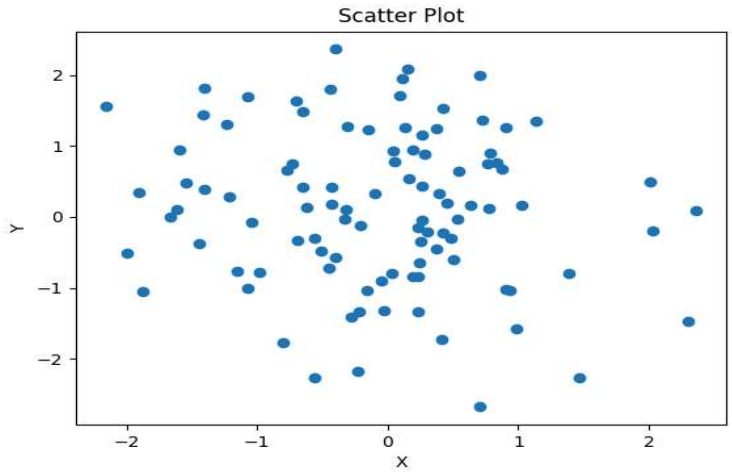


Q4(B):Write a python program to demonstrate how to draw a scatter plot using mathplotlib.

Soln:

```
import matplotlib.pyplot as plt
import numpy as npx = np.random.randn(100)
y = np.random.randn(100)
plt.scatter(x, y)
plt.title('Scatter Plot')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```

Output:

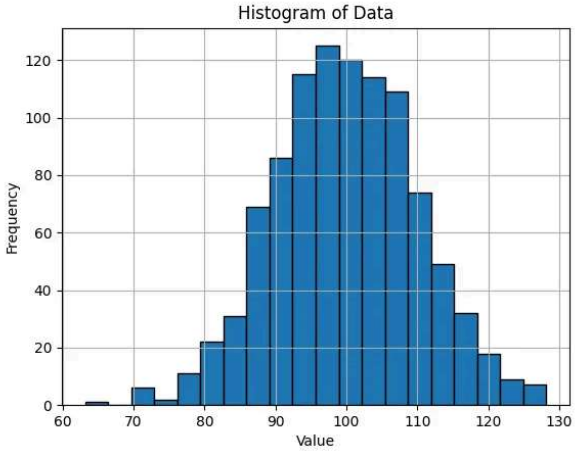


Q5(A):Write a python program to demonstrate how to draw a histogram plot using mathplot lib.

Soln:

```
import matplotlib.pyplot as plt
import numpy as npdata = np.random.normal(100, 10, 1000)
plt.hist(data, bins=20, edgecolor='black')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Histogram of Data')
plt.grid(True)
plt.show()
```

Output:

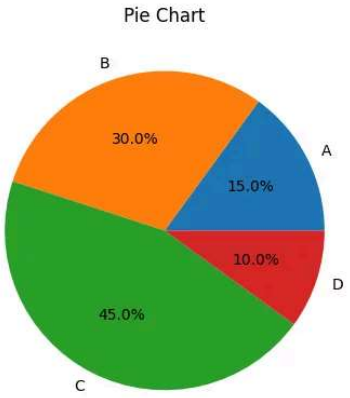


Q5(B):Write a python program to demonstrate how to draw a pie chart using mathplot lib.

Soln:

```
import matplotlib.pyplot as pltlabels = ['A', 'B', 'C', 'D']
sizes = [15, 30, 45, 10]
plt.pie(sizes, labels=labels, autopct="%1.1f%%")
plt.title("Pie Chart")
plt.show()
```

Output:



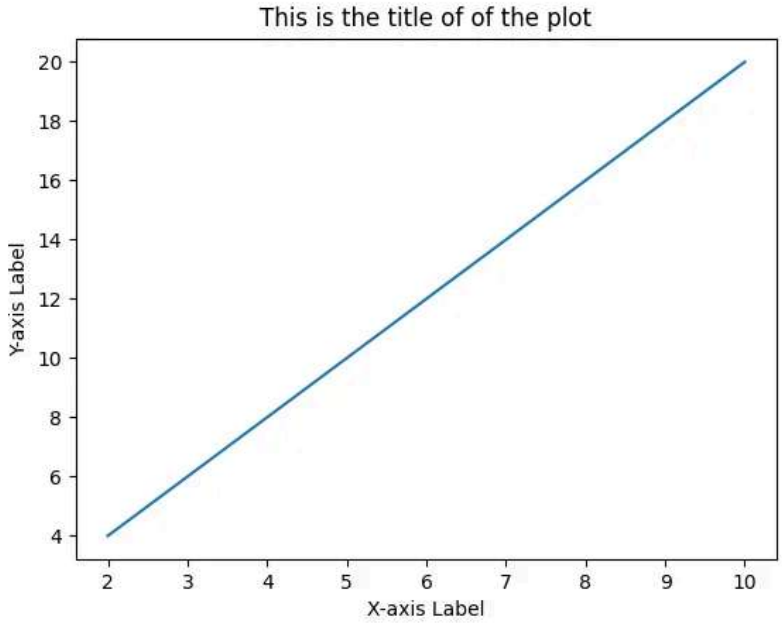
Q6(A):Write a program to illustrate linear plotting using matplolib.

Soln:

```
import matplotlib.pyplot as plt
import numpy as npX = np.array([2, 4, 6, 8, 10])
Y = X * 2
plt.plot(X, Y)
plt.xlabel("X-axis Label")
```

```
plt.ylabel("Y-axis Label")
plt.title("This is the title of of the plot")
plt.show()
```

Output:



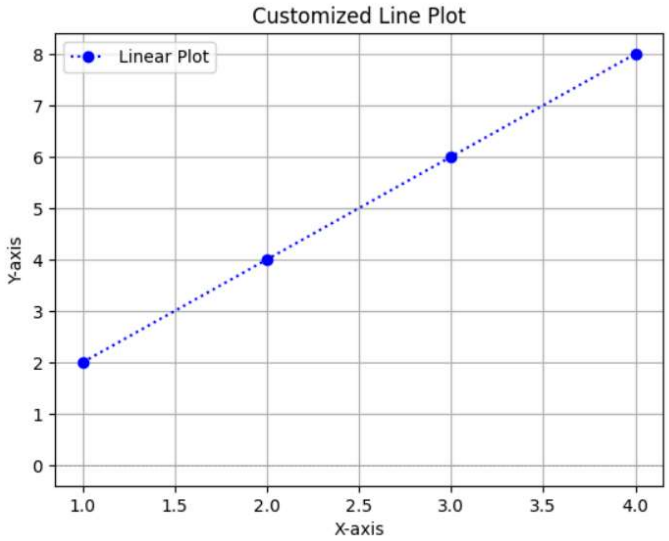
Q6(B):Write a program to illustrate linear plotting using line formatting using matplotlib.

Soln:

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

# Generate sample data
x = np.array([1, 2, 3, 4])
y = x * 2

# Create a simple line plot
plt.plot(x, y, linestyle=':', marker='o', color='b', label='Linear Plot')

# Add labels and title
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Customized Line Plot")

# Customize line style and color
```

```
plt.grid(True)
plt.axhline(y=0, color='gray', linestyle='--', linewidth=0.5)

# Display the legend
plt.legend()

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

Output:

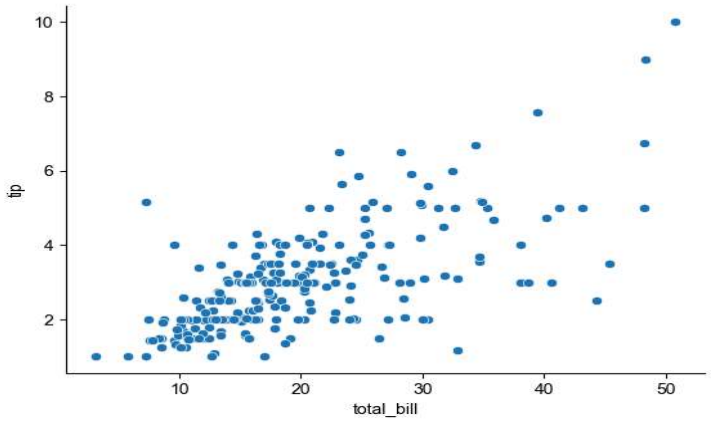


Q7:Write a python program which explains use of customizing use of seabourn with aesthetic functions .

Soln:

```
import seaborn as sns
import matplotlib.pyplot as plttips = sns.load_dataset("tips")
sns.scatterplot(x="total_bill", y="tip", data=tips)
sns.set_style("whitegrid")
sns.set_palette("Set2")
sns.despine()
plt.show()
```

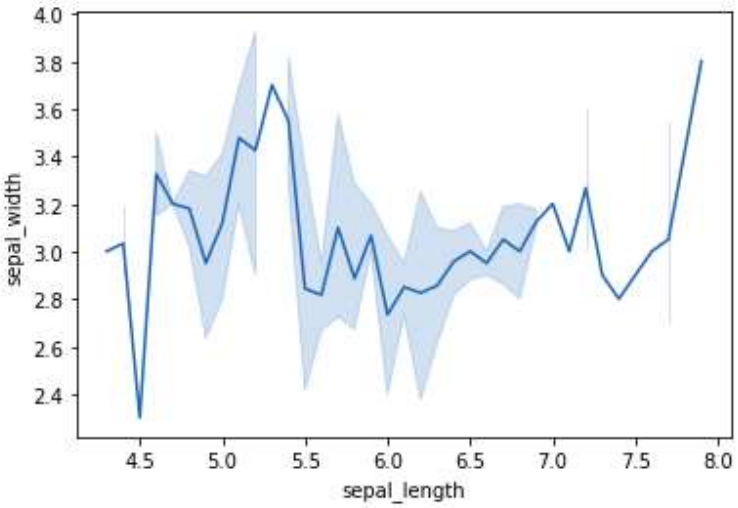
Output:



Or

```
import seaborn as sns
data = sns.load_dataset("iris")
sns.lineplot(x="sepal_length", y="sepal_width", data=data)
```

Output:



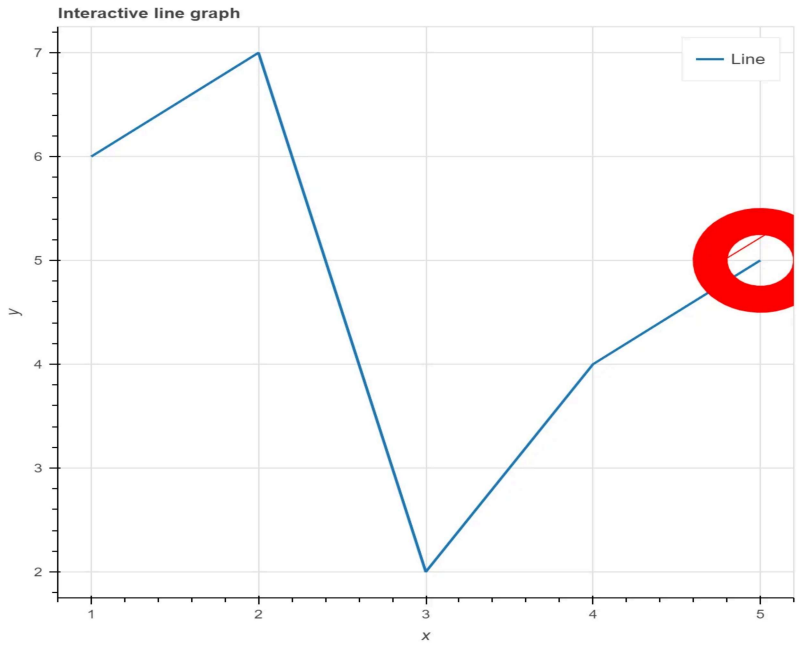
Q8:Write a python program to explain working with bokeh line graph using annotations and legends.

(A):Write a python program for plotting different types of plots using bokeh.

Soln:

```
from bokeh.plotting import figure, show
x = [1, 2, 3, 4, 5]
y = [6, 7, 2, 4, 5]
p = figure(title="Interactive line graph", x_axis_label='x', y_axis_label='y')
p.line(x, y, legend_label="Line", line_width=2)
p.annular_wedge(x=5, y=5, inner_radius=0.2, outer_radius=0.4, start_angle=45,
end_angle=135, line_color="red", fill_color="red")
```

show(p)

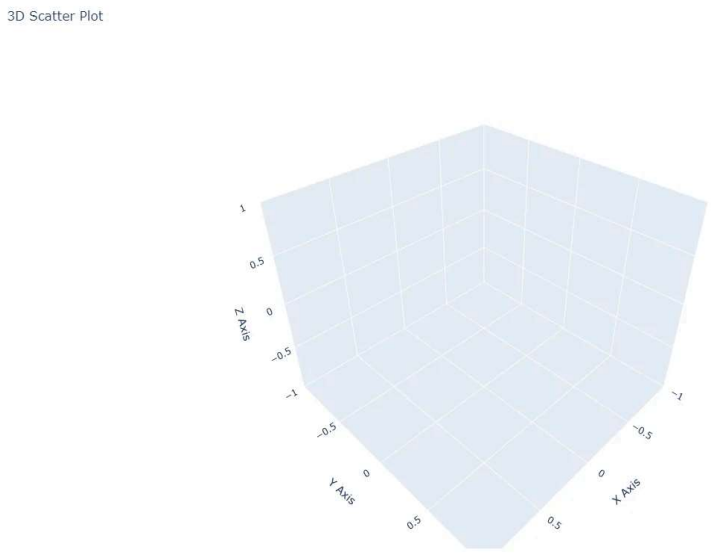


Q9:Write a python program to draw 3-D plots using plotly libraries.

Soln:

```
import plotly.graph_objects as go
import numpy as npx = np.linspace(0, 10, 100)
y = np.linspace(0, 10, 100)
z = np.random.randn(100, 100)
fig = go.Figure(data=[go.Scatter3d(x=x, y=y, z=z, mode='markers')])
fig.update_layout(title='3D Scatter Plot', scene=dict(xaxis_title='X Axis',
yaxis_title='Y Axis', zaxis_title='Z Axis'))
fig.show()
```

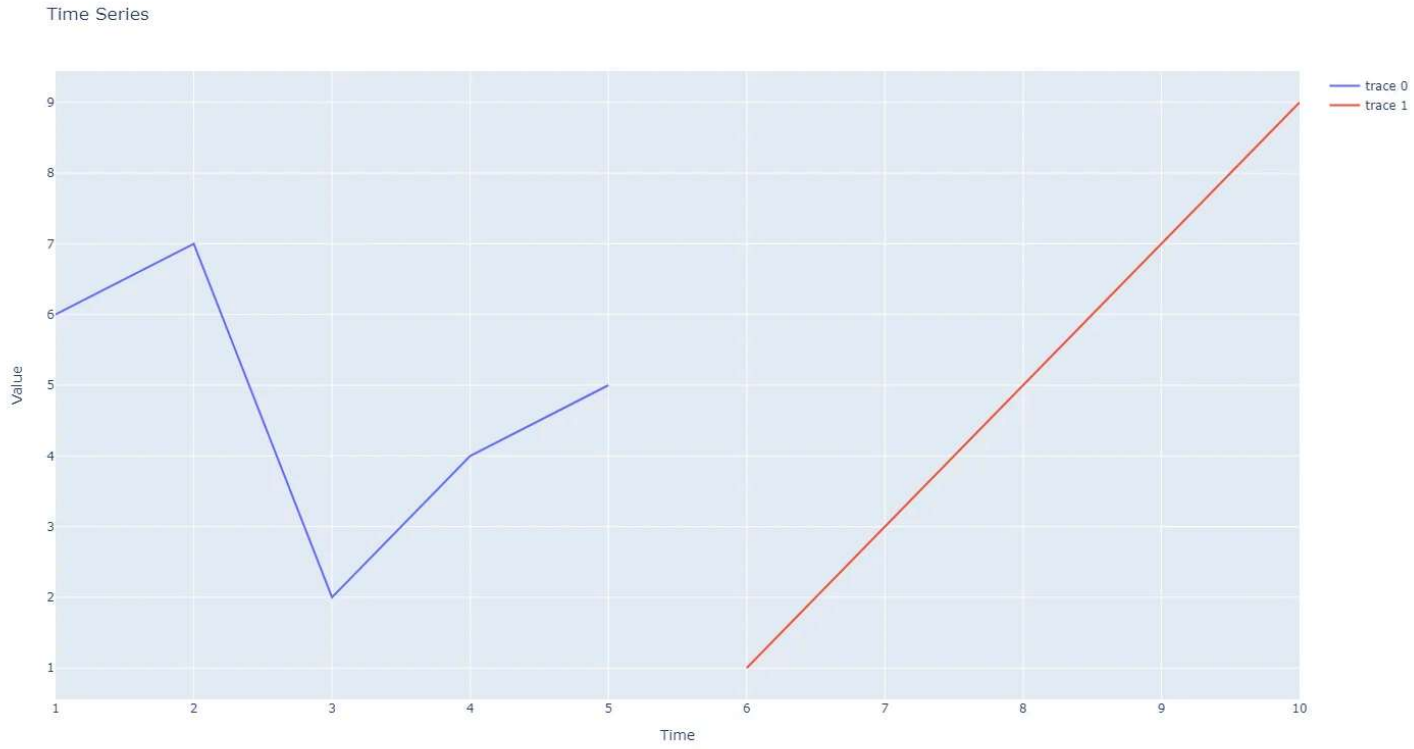
Output:



Q10(A):Write a python program to draw time series using plotly libraries.
Soln:

```
import plotly.graph_objects as godata = [
{'x': [1, 2, 3, 4, 5], 'y': [6, 7, 2, 4, 5]},
{'x': [6, 7, 8, 9, 10], 'y': [1, 3, 5, 7, 9]}
]
fig = go.Figure()
for i in range(len(data)):
fig.add_trace(go.Scatter(x=data[i]['x'], y=data[i]['y'], mode='lines'))
fig.update_layout(title='Time Series', xaxis_title='Time', yaxis_title='Value')
fig.show()
```

Output:



Q10(B):Write a python program for creating maps using plotly libraries.
Soln:

```
import plotly.express as pxdf = px.data.election()
geojson = px.data.election_geojson()fig = px.choropleth(df, geojson=geojson,
color="Bergeron",
locations="district", featureidkey="properties.district",
projection="mercator"
)
fig.update_geos(fitbounds="locations", visible=True)
```

```
fig.update_layout(margin={"r": 0, "t": 0, "l": 0, "b": 0})
fig.show()
```

Output: