In [1]:

```
import bokeh.io
import bokeh.plotting
bokeh.io.output_notebook()
```

Loading BokehJS ...

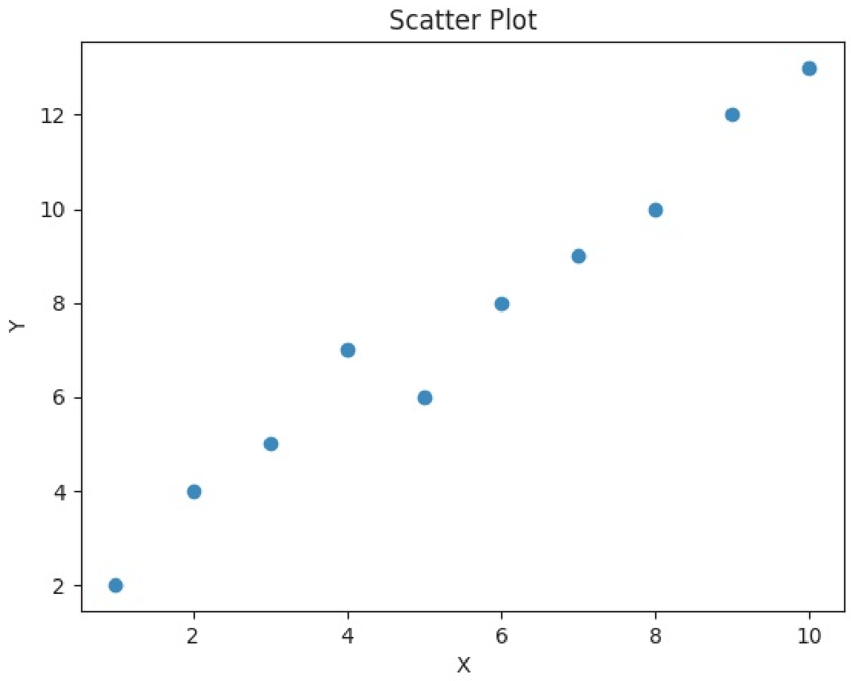
In [2]:

```
from bokeh.sampledata.iris import flowers
from bokeh.plotting import figure, output_file, show
from bokeh.plotting import figure, show
from bokeh.models import ColumnDataSource
from bokeh.palettes import Category10
from bokeh.models import ColorBar, LinearColorMapper
from bokeh.palettes import Viridis256
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

MATPLOTLIB ASSIGNMENT # 1. Create a scatter plot using Matplotlib to visualize the relationship between two arrays,x and y for given data. x= [1,2,3,4,5,6,7,8,9,10] y= [2,4,5,7,6,8,9,10,12,13]

In [3]:

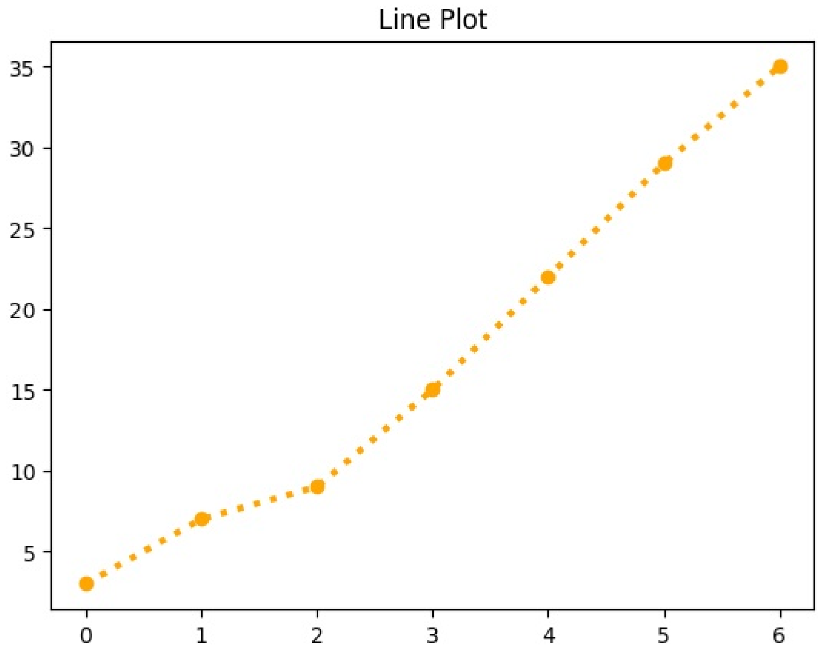
```
x=[1,2,3,4,5,6,7,8,9,10]
y=[2,4,5,7,6,8,9,10,12,13]

plt.scatter(x,y)
plt.title("Scatter Plot")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```



2. Generate a line plot to visualize the trend of values for given data. data = np.array([3,7,9,15,22,29,35])

In [4]:

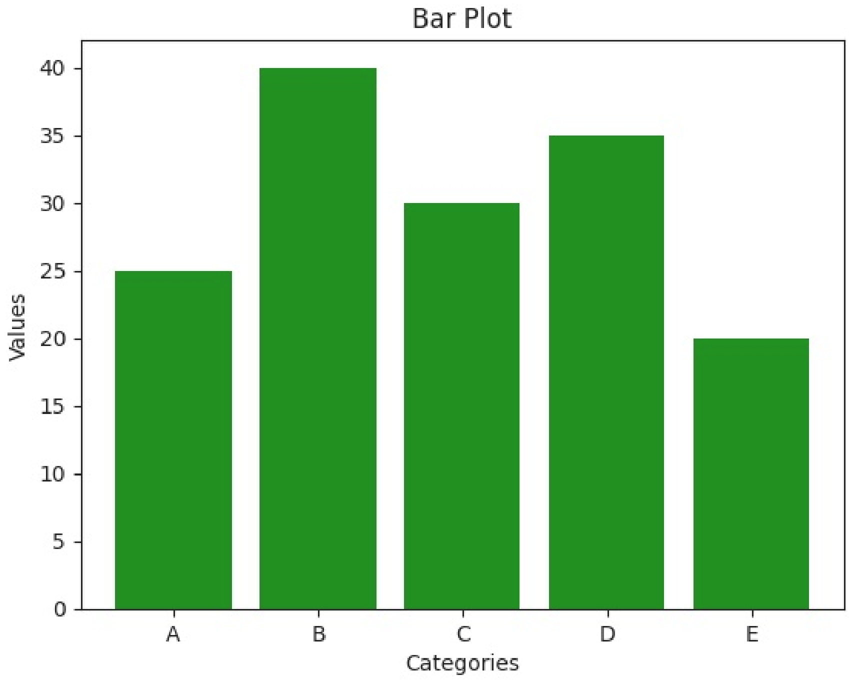
```
data= np.array([3,7,9,15,22,29,35])
plt.plot([3,  7,  9, 15, 22, 29, 35],color="orange",marker="o",linestyle= ':',linewidth=3)
plt.title('Line Plot')

plt.show()
```



# 3. Display a bar chart to represent the frequency of each item in the given array categories. categories = ['A','B','C','D','E'] values = [25,40,30,35,20]

In [5]:

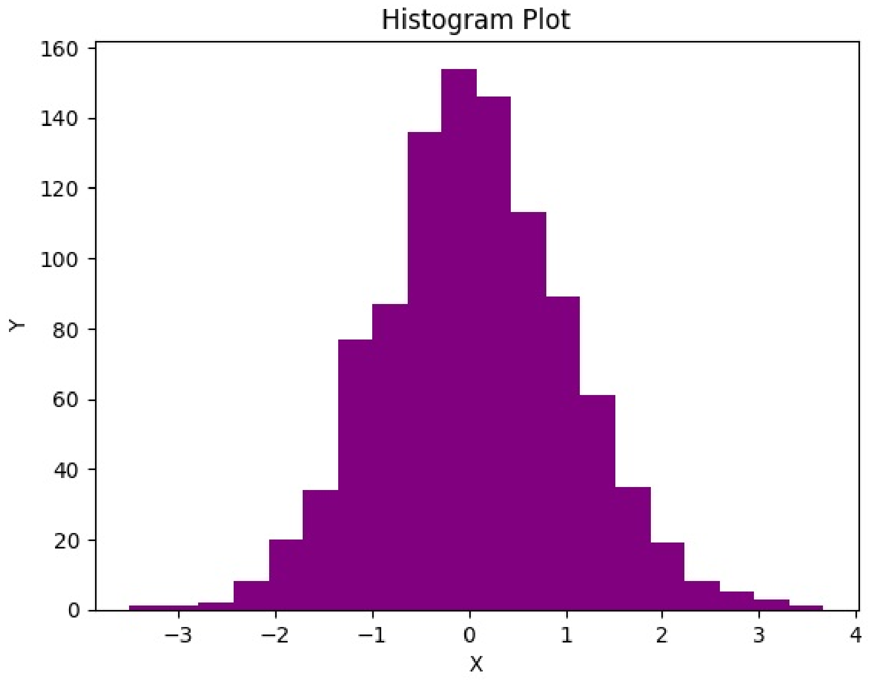
```
categories = ['A','B','C','D','E']
values = [25,40,30,35,20]

plt.bar(categories,values, color= 'green')
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Plot')

plt.show()
```



# 4. Create a histogram to visualize the distribution of values in the array data. data = np.random.normal(0,1,1000)

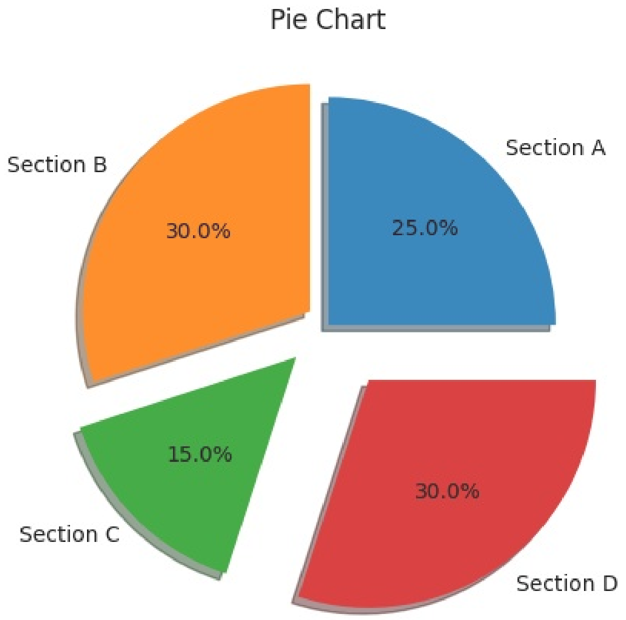
In [6]:

```
data = np.random.normal(0,1,1000)
plt.hist(data,color="purple",bins=20)
plt.title('Histogram Plot')
plt.xlabel('X')
plt.ylabel('Y')

plt.show()
```



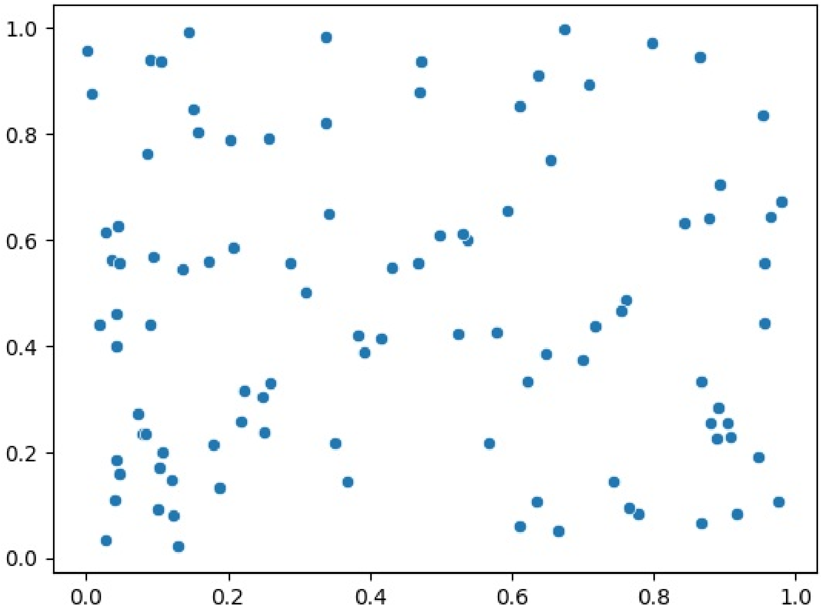
# 5. Show the pei chart to represent the percentage distribution of different sections in the array sections. sections = ['Section A','Section B','Section C','Section D'] sizes = [25,30,15,30]

```
In [7]: explode= (0.0,0.1,0.2,0.3)

        sections= ['Section A','Section B','Section C','Section D']
        sizes= [25,30,15,30]

        plt.pie(sizes,labels = sections,explode=explode,autopct= '%1.1f%%', shadow= True)
        plt.title("Pie Chart")

        plt.show()
```



SEABORN ASSIGNMENT #1.Create a scatter plot to visualize the relationship between two variables, by generating a synthetic dataset.

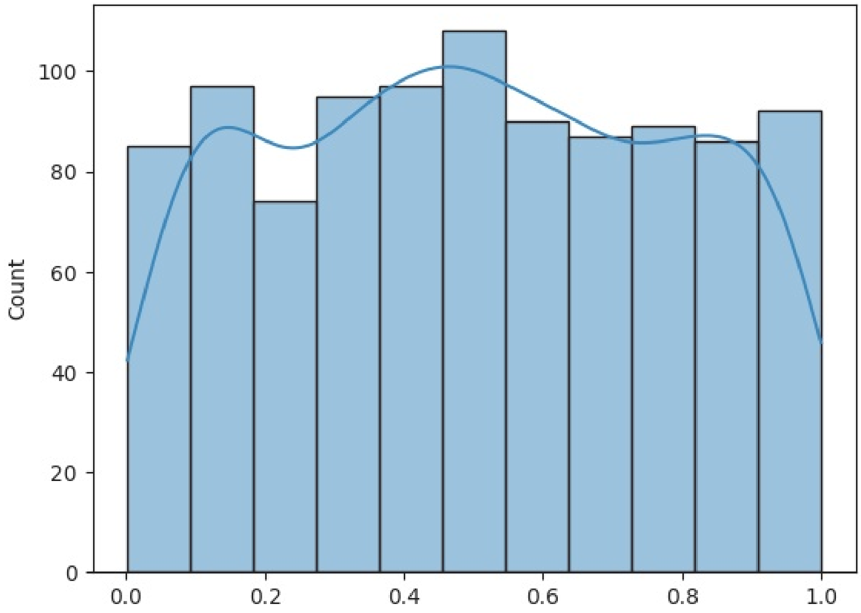
```
In [9]: x , y = np.random.rand(2, 100)
        sns.scatterplot(x = x, y = y)
        plt.show()
```



#2. Generate a dataset if random numbers , visualize the distribution of a numerical variable.

In [10]:

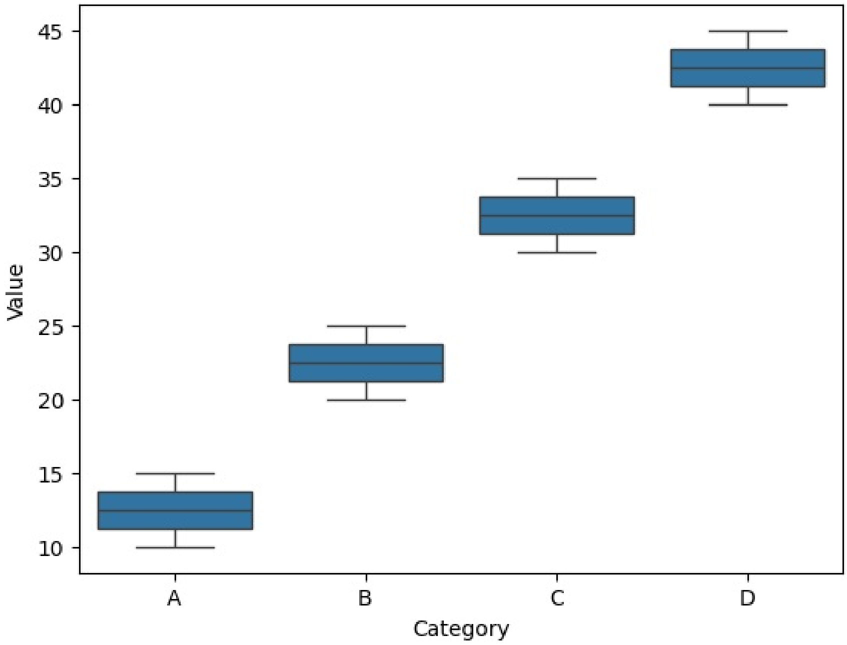
```
sns.histplot(np.random.rand(1000),kde = True)
plt.show()
```



#3. Create a dataset representing categories and their corresponding valuesl Compare different categories based on numerical values.

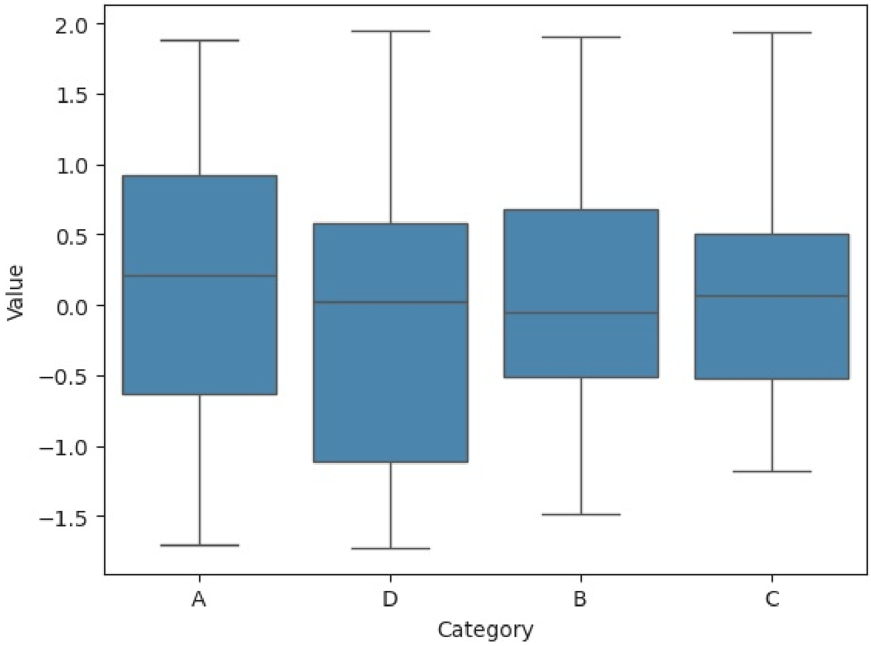
In [11]:

```
data = {'Category':['A','B','C','D','A','B','C','D'],
        'Value':[10,20,30,40,15,25,35,45]}
df = pd.DataFrame(data)
sns.boxplot(x='Category',y='Value',data=df)
plt.show()
```



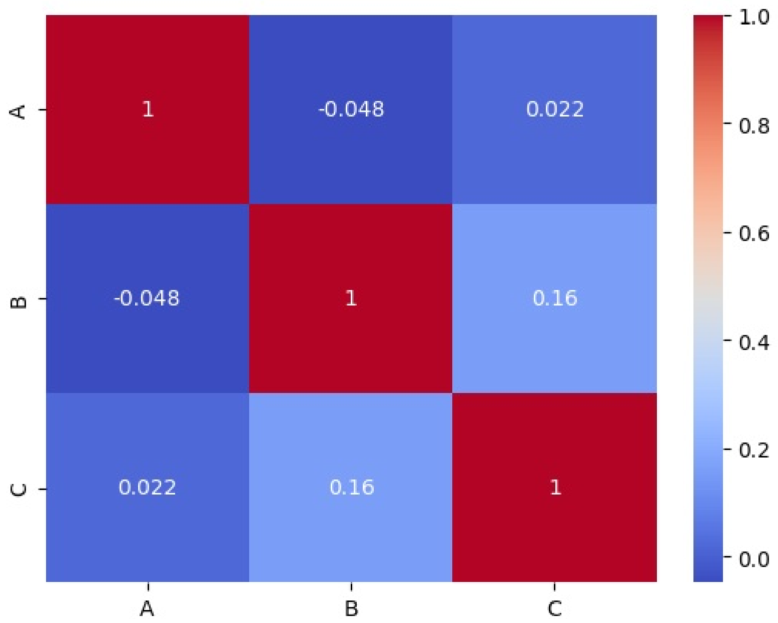
#4.Generate a dataset with categories and numerical values. Visualize the distribution of a numerical variable across different categories.

```
In [12]: np.random.seed(0)
         df = pd.DataFrame({
             'Category': np.random.choice(['A', 'B', 'C', 'D'], 100),
             'Value': np.random.randn(100)})

         sns.boxplot(x='Category', y='Value', data=df)
         plt.show()
```



#5.Generate a synthetical dataset with correlated features.Visualize the correlation matrix of a dataset using a heatmap

```
In [13]: df = pd.DataFrame(np.random.rand(100, 3), columns=['A', 'B', 'C'])

         sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
         plt.show()
```



PLOTLY ASSIGNMENT # 1.Using the given dataset,to generate a 3d scatter plot to visualize the distribition of data points in a there dimensional space.

In [14]:

```
np.random.seed(30)
data = {'X':np.random.uniform(-10,10,300),'Y':np.random.uniform(-10,10,300),'Z':np.random.uniform(-10,10,300)}
df = pd.DataFrame(data)
df
```

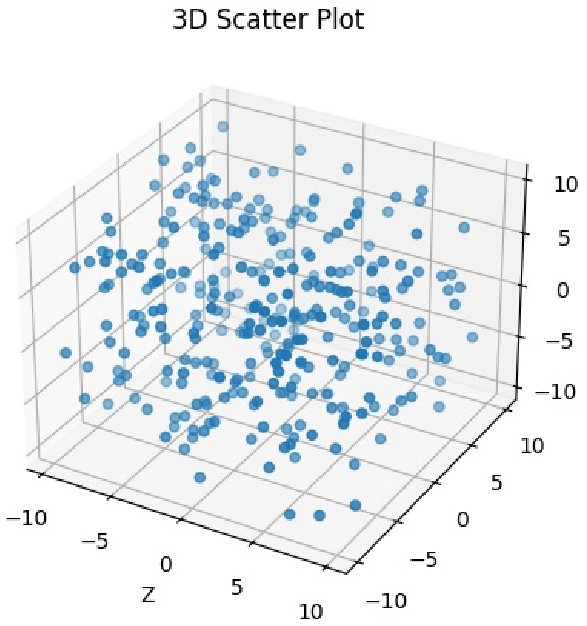
Out[14]:

	X	Y	Z
0	2.882871	-7.376076	6.925787
1	-2.385030	-5.950714	-8.410773
2	3.260958	0.853469	-6.647725
3	-6.726985	-6.602615	-3.725201
4	9.252156	-4.791792	-0.580081
...	...	...	...
295	-0.369458	-6.543074	9.941758
296	-3.675668	-6.671420	-2.019459
297	-6.393715	-6.040683	1.774010
298	-0.007816	9.738364	-1.721501
299	-4.623348	7.697305	1.604021

300 rows × 3 columns

In [15]:

```
fig= plt.figure()
ax= fig.add_subplot(projection='3d')
ax.scatter(df['X'],df['Y'],df['Z'],)
ax.set_xlabel('X')
ax.set_xlabel('Y')
ax.set_xlabel('Z')
plt.title('3D Scatter Plot')

plt.show()
```



# 2. Using the student Grades, create a violin plot to display the distribution of scores across different grade categories.

In [16]:

```python
np.random.seed(15)
data = {'Grade':np.random.choice(['A','B','C','D','F'],200),
        'Score':np.random.randint(50,100,200)}
df = pd.DataFrame(data)
df
```

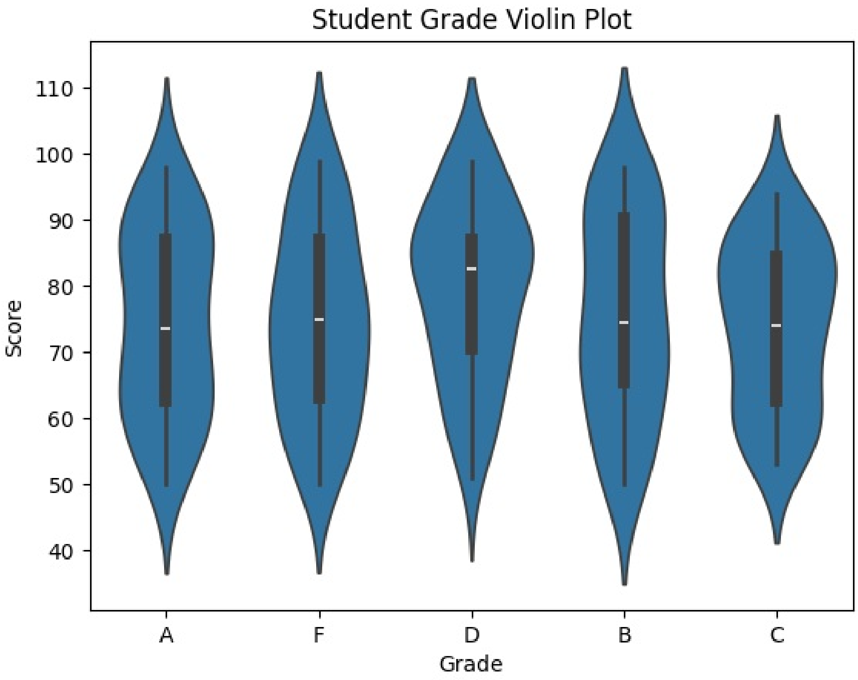
Out[16]:

	Grade	Score
0	A	89
1	F	50
2	A	72
3	F	66
4	D	87
...	...	...
195	F	78
196	D	84
197	A	93
198	D	67
199	C	56

200 rows × 2 columns

In [17]:

```python
sns.violinplot(x='Grade',y='Score',data=df)
plt.title('Student Grade Violin Plot')
plt.xlabel('Grade')
plt.ylabel('Score')

plt.show()
```



# 3. Using the sales data,generate a heatmap to visualize the variation in sales across different months and days.

In [18]:

```
np.random.seed(20)
data = {'Month':np.random.choice(['Jan','Feb','Mar','Apr','May'],100),
      'Day':np.random.choice(range(1,31),100),
      'Sales':np.random.randint(1000,5000,100)}
df = pd.DataFrame(data)
df
```

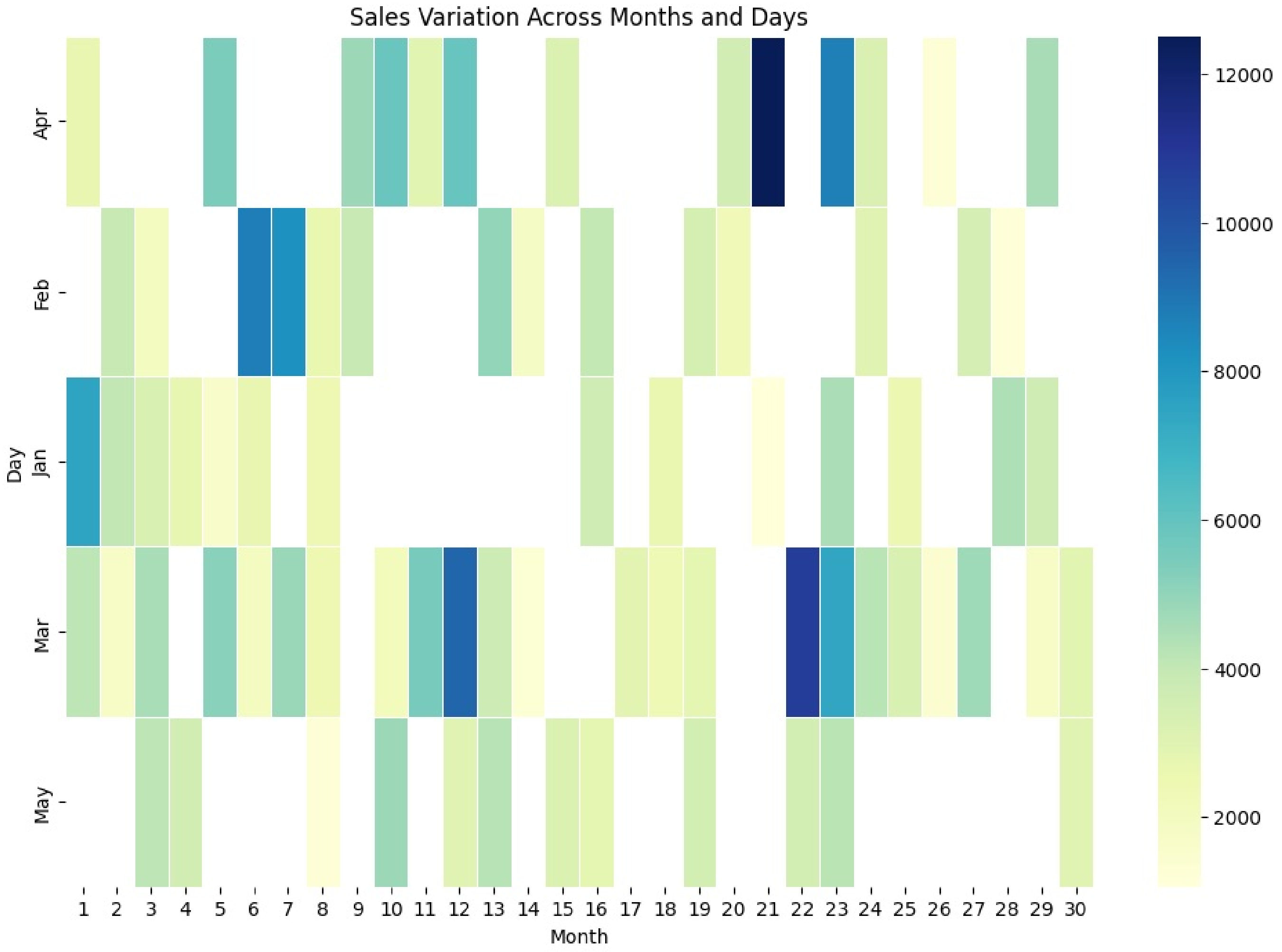
Out[18]:

	Month	Day	Sales
0	Apr	23	4835
1	Mar	10	2099
2	May	10	2094
3	Mar	22	4809
4	Feb	3	1969
...	...	...	...
95	May	19	1375
96	Mar	12	3909
97	Feb	20	2264
98	May	16	1548
99	Feb	9	3893

100 rows × 3 columns

In [19]:

```
heatmap_data = df.pivot_table(values='Sales', index='Month',columns='Day',aggfunc='sum')

df['Month'] = pd.Categorical(df['Month'],categories = ['Jan','Feb','Mar','Apr','May'], ordered=True)
df['Day'] = pd.Categorical(df['Day'],categories=range(1,32),ordered=True)

plt.figure(figsize=(12,8))
sns.heatmap(heatmap_data,annot=False,cmap='YlGnBu',linewidth=.5)
plt.title('Sales Variation Across Months and Days')
plt.xlabel('Month')
plt.ylabel('Day')

plt.show()
```



# 4.Using the given x and y data,generate a 3D surface plot to visualize the function z = sin(x^2 + y^2)

In [20]:

```
x = np.linspace(-5,5,100)
y = np.linspace(-5,5,100)
x,y = np.meshgrid(x,y)
z = np.sin(np.sqrt(x**2 + y**2))

data = {'X':x.flatten(),
        'Y':y.flatten(),
        'Z':z.flatten(),}
df = pd.DataFrame(data)
```

In [21]:

```
df
```

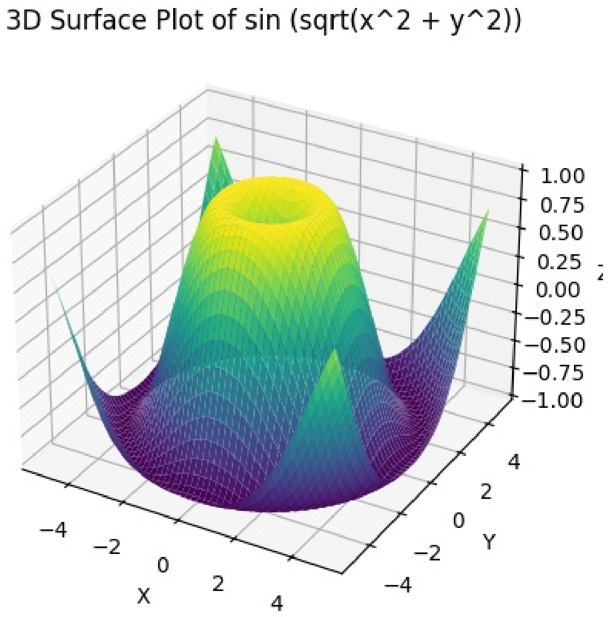
Out[21]:

	X	Y	Z
0	-5.00000	-5.0	0.708861
1	-4.89899	-5.0	0.656992
2	-4.79798	-5.0	0.602401
3	-4.69697	-5.0	0.545474
4	-4.59596	-5.0	0.486601
...	...	...	...
9995	4.59596	5.0	0.486601
9996	4.69697	5.0	0.545474
9997	4.79798	5.0	0.602401
9998	4.89899	5.0	0.656992
9999	5.00000	5.0	0.708861

10000 rows × 3 columns

In [22]:

```
fig= plt.figure()
ax= fig.add_subplot(111,projection='3d')
surf= ax.plot_surface(x, y, z, cmap='viridis')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.title('3D Surface Plot of sin (sqrt(x^2 + y^2))')
plt.show()
```



# 5.Using the given dataset,create a bubble chart to represent each country's population (y-axis), GDP(x-axis), and bubble size proportional to the population

In [23]:

```
np.random.seed(25)
data = {'Country':['USA','Canada','UK','Germany','France'],
        'Population':np.random.randint(100,1000,5),
        'GDP':np.random.randint(500,2000,5)}
df = pd.DataFrame(data)
```

In [24]:

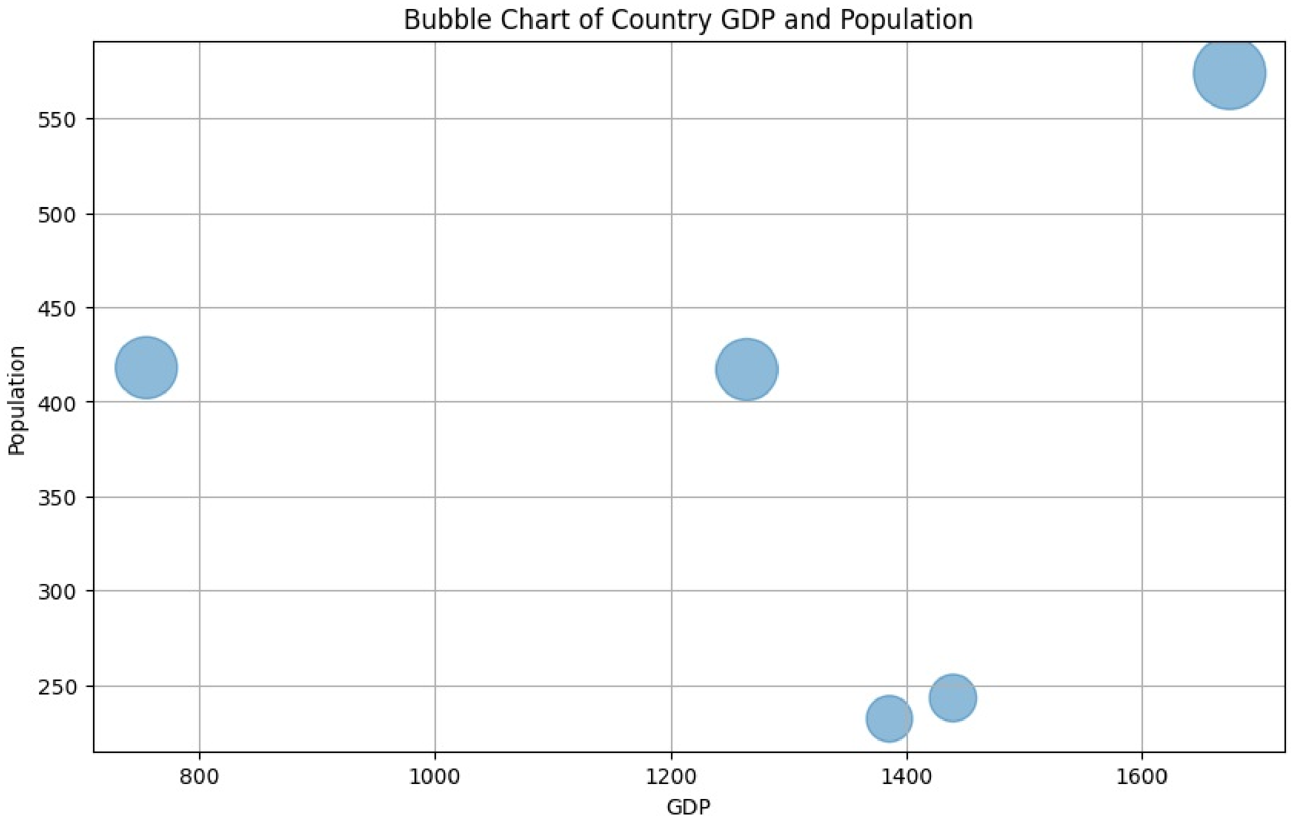
```
df
```

Out[24]:

	Country	Population	GDP
0	USA	232	1386
1	Canada	418	755
2	UK	574	1675
3	Germany	243	1440
4	France	417	1265

In [25]:

```
plt.figure(figsize=(10,6))
plt.scatter(df['GDP'],df['Population'],s=df['Population']*2,alpha=0.5)

plt.xlabel('GDP')
plt.ylabel('Population')
plt.title('Bubble Chart of Country GDP and Population')
plt.grid(True)

plt.show()
```



BOKEH ASSIGNMENT # 1. Create a Bokeh plot displaying a sine wave, Set x-values from 0 to 10 and y-values as the sine of x.

In [31]:
```
x = np.linspace(0,10,100)
y = np.sin(x)

p = figure(title = 'Sine Wave', x_axis_label = "x", y_axis_label = "sin(x)")
p.line(x,y, line_width = 2, color = "orange")

show(p)
```

# 2. Creat a Bokeh scatter plot using randomly generated x and y values. Use different sizes and colors for the markers based on the 'size' and 'colors' columns.

In [32]:
```
num_points = 100
sizes = np.random.randint(5, 20, num_points)
colors = np.random.choice(['red','blue','green','yellow'], num_points)
x_values = np.random.rand(num_points)
y_values = np.random.rand(num_points)

p= figure(title = "Scatter Plot with Variable Size and Color")
p.scatter(x_values, y_values, size = sizes, color = colors, alpha = 0.6)

p.xaxis.axis_label = "X-axis"
p.yaxis.axis_label = "Y-axis"

show(p)
```

# 3. Generate a Bokeh bar chart representing the counts of different fruits using the following dataset. fruits = ['Apples', 'Oranges', 'Banana', 'Pears'] counts = [20, 25, 30, 35]

In [33]:
```
fruits = ['Apples', 'Oranges', 'Banana', 'Pears']
counts = [20, 25, 30, 35]
source = ColumnDataSource(data=dict(fruits=fruits, counts=counts))

p = figure(x_range=fruits, height=350, title= "Fruit Counts", toolbar_location= None, tools="")

p.vbar(x='fruits', top='counts', width=0.9, source=source)

p.xgrid.grid_line_color= None
p.yaxis.axis_label = "Count"
p.xaxis.axis_label = "Fruit"

show(p)
```

# 4. Create a Bokeh histogram to visualize the distribution of the given data. data_hist = np.random.randn(1000) hist, edges = np.histogram(data_hist, bins=30)

In [34]:
```
data_hist = np.random.randn(1000)
hist, edges = np.histogram(data_hist, bins=30)
source = ColumnDataSource(data=dict(x=edges[:-1], top=hist))
```

```
p = figure(title="Histogram of Random Data", x_axis_label="Value", y_axis_label="Frequency")
p.quad(bottom=0, top='top', left='x',right='x',source=source, fill_color="blue", line_color="green", width=7)

show(p)
```

# 5. Create a Bokeh heatmap using the provided dataset. data_heatmap = np.random.rand(10,10) x = np.linspace(0, 1, 10) y = np.linspace(0, 1, 10) xx, yy = np.meshgrid(x, y)

In [35]:

```
data_heatmap = np.random.rand(10, 10)
x = np.linspace(0, 1, 10)
y = np.linspace(0, 1, 10)
xx, yy = np.meshgrid(x, y)


color_mapper = LinearColorMapper(palette=Viridis256, low=data_heatmap.min(), high=data_heatmap.max())
p.rect(x='x', y='y', width=0.1, height=0.1, source={'x': xx.flatten(), 'y': yy.flatten(), 'value': data_heatmap

color_bar = ColorBar(color_mapper=color_mapper, label_standoff=12)

show(p)
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js