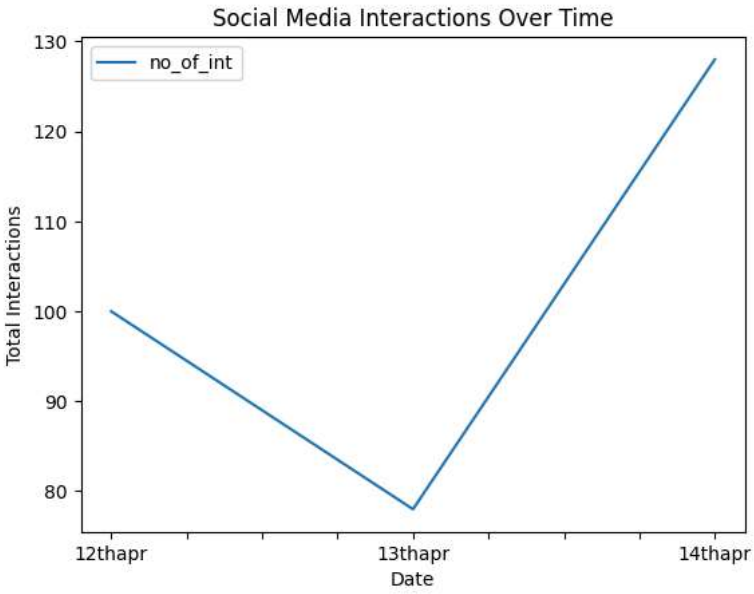
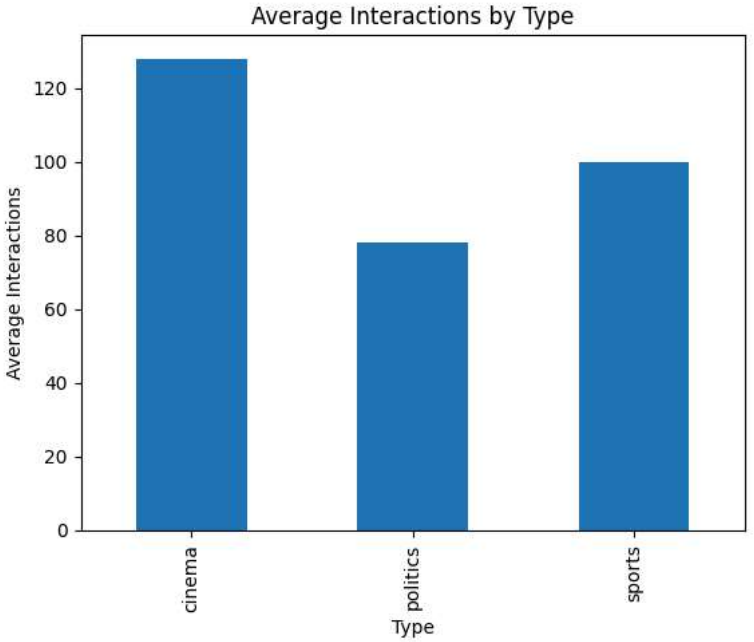
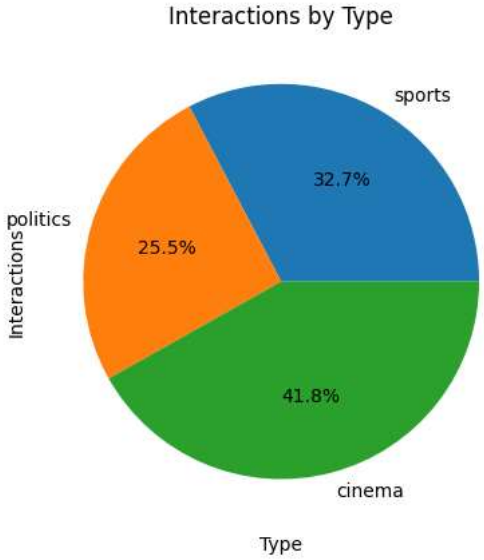
```
import pandas as pd
import numpy as np
s_r=pd.DataFrame()
s_r["name"]=["abi","mani","sunder","rajan"]
print(s_r)
object=pd.Series([1,2,3,4])
s_r["object"]=object
print(s_r)
print(s_r.shape)
print(s_r.info())
print(end="\n\n")
print(s_r.describe())
```

```
     name
0     abi
1    mani
2  sunder
3   rajan
     name  object
0     abi       1
1    mani       2
2  sunder       3
3   rajan       4
(4, 2)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   name    4 non-null      object
 1   object  4 non-null      int64
dtypes: int64(1), object(1)
memory usage: 192.0+ bytes
None


         object
count  4.000000
mean   2.500000
std    1.290994
min    1.000000
25%    1.750000
50%    2.500000
75%    3.250000
max    4.000000
```

```
import matplotlib.pyplot as plt
import pandas as pd

analytics=pd.DataFrame()

analytics["Date"] = ["12thapr","13thapr","14thapr"]
analytics["Type"]= ["sports","politics","cinema"]
analytics["no_of_int"]= [100,78,128]
analytics["Likes"] = [48,17,38]
analytics["Shares"] = [17,19,52]
analytics["Comments"] = [42,52,60]

analytics.plot(x="Date", y="no_of_int")
plt.xlabel("Date")
plt.ylabel("Total Interactions")
plt.title("Social Media Interactions Over Time")
plt.show()

analytics.groupby("Type")["no_of_int"].mean().plot(kind="bar")
plt.xlabel("Type")
plt.ylabel("Average Interactions")
plt.title("Average Interactions by Type")
plt.show()

plt.pie(analytics["no_of_int"],labels=analytics["Type"],autopct="%1.1f%%")
plt.xlabel("Type")
plt.ylabel("Interactions")
plt.title("Interactions by Type")
plt.show()
```


Social Media Interactions Over Time
no_of_int
Total Interactions
130
120
110
100
90
80
12thapr
13thapr
14thapr
Date


Average Interactions by Type
Average Interactions
120
100
80
60
40
20
0
cinema
politics
sports
Type


Interactions by Type
sports
32.7%
politics
25.5%
41.8%
cinema
Interactions
Type

```
# prompt: various pandas usage

# Create a DataFrame from a dictionary
data = {'Name': ['Alice', 'Bob', 'Claire'], 'Age': [25, 30, 28]}
df = pd.DataFrame(data)

# Print the DataFrame
print(df)

# Create a Series from a list
data = [1, 2, 3, 4, 5]
series = pd.Series(data)

# Print the Series
print(series)

# Accessing elements
print(df['Name'][0])
print(series[2])

# Adding a new column
df['Occupation'] = ['Student', 'Engineer', 'Doctor']
print(df)

# Deleting a column
del df['Occupation']
print(df)

# Sorting
df.sort_values('Age', inplace=True)
print(df)


# Filtering
filtered_df = df[df['Age'] > 28]
print(filtered_df)

# Applying functions
def double(x):
  return x * 2

df['Age'] = df['Age'].apply(double)
print(df)
```

```
      Name  Age
0   Alice   25
1     Bob   30
2  Claire   28
0    1
1    2
2    3
3    4
4    5
dtype: int64
Alice
3
     Name  Age Occupation
0   Alice   25    Student
1     Bob   30   Engineer
2  Claire   28     Doctor
     Name  Age
0   Alice   25
1     Bob   30
2  Claire   28
     Name  Age
0   Alice   25
2  Claire   28
1     Bob   30
  Name  Age
1  Bob   30
     Name  Age
0   Alice   50
2  Claire   56
1     Bob   60
```

```
# prompt: dataframe shortcuts

# Create a DataFrame from a dictionary
data = {'Name': ['Alice', 'Bob', 'Claire'], 'Age': [25, 30, 28]}
df = pd.DataFrame(data)

# Create a Series from a list
data = [1, 2, 3, 4, 5]
series = pd.Series(data)

# Accessing elements
print(df['Name'][0])  # Access the first element of the 'Name' column
print(series[2])  # Access the third element of the Series

# Adding a new column
df['Occupation'] = ['Student', 'Engineer', 'Doctor']

# Deleting a column
del df['Occupation']

# Sorting
df.sort_values('Age', inplace=True)  # Sort by the 'Age' column in ascending order

# Filtering
filtered_df = df[df['Age'] > 28]  # Filter rows where 'Age' is greater than 28

# Applying functions
def double(x):
  return x * 2

df['Age'] = df['Age'].apply(double)  # Apply the 'double' function to the 'Age' column

# Other shortcuts
  # Show information about the DataFrame, such as the data types and number of non-null values
```

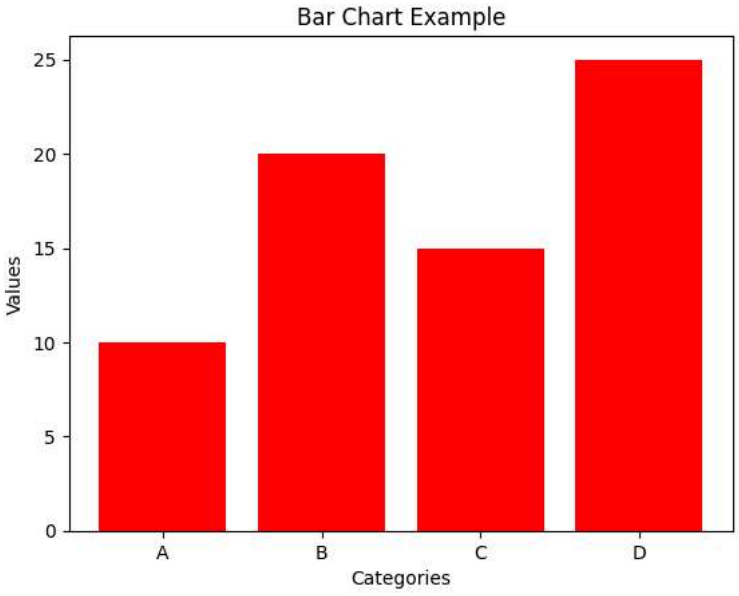
```
Alice
3
```

	Age
count	3.000000
mean	55.333333
std	5.033223
min	50.000000
25%	53.000000
50%	56.000000
75%	58.000000
max	60.000000

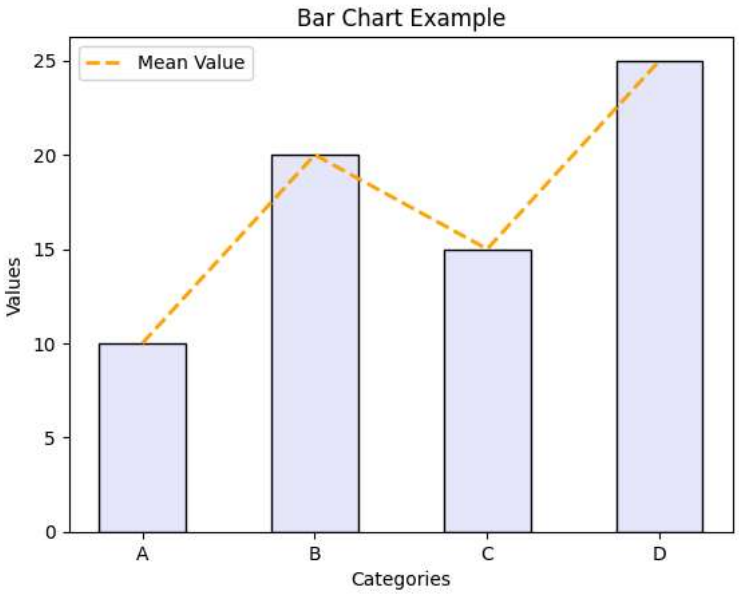
```
#df.head(2)  # Show the first few rows of the DataFrame
#df.tail(2)  # Show the last few rows of the DataFrame
#df.describe()  # Show descriptive statistics of the DataFrame
#df.info()
```

	Age
count	3.000000
mean	55.333333
std	5.033223
min	50.000000
25%	53.000000
50%	56.000000
75%	58.000000
max	60.000000

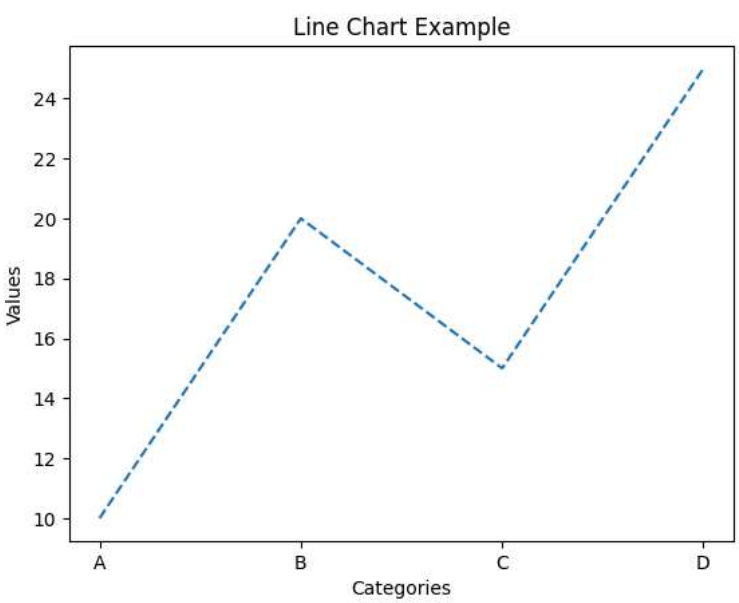
```
import matplotlib.pyplot as plt

# Data
categories = ['A', 'B', 'C', 'D']
values = [10, 20, 15, 25]

# Create bar chart
plt.bar(categories, values,color="red")
# Add labels and title
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Chart Example')

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



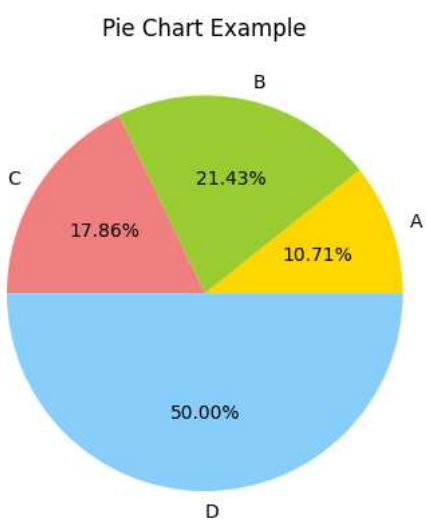

```
import matplotlib.pyplot as plt

# Data
categories = ['A', 'B', 'C', 'D']
values = [10, 20, 15, 25]

# Customize attributes
bar_color = 'lavender'
line_color = 'orange'
line_width = 2
bar_width = 0.5

plt.plot(categories,values,color=line_color,linewidth=line_width, linestyle='--', label='Mean Value')
plt.bar(categories, values, color=bar_color, width=bar_width, edgecolor='black')

# Add a horizontal line
#plt.plot()

# Add labels and title
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Chart Example')

# Add legend
plt.legend()

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




```
import matplotlib.pyplot as plt

# Data
categories = ['A', 'B', 'C', 'D']
values = [10, 20, 15, 25]

# Create line chart
plt.plot(categories, values, linestyle='--')

# Add labels and title
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Line Chart Example')

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

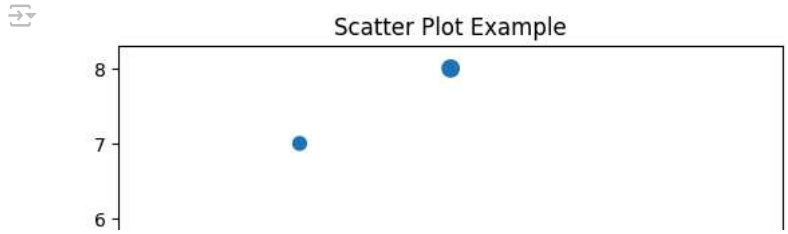


```
import matplotlib.pyplot as plt

# Data
labels = ['A', 'B', 'C', 'D']
sizes = [15, 30, 25, 70]
colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue']

# Create pie chart
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.2f%%')

# Equal aspect ratio ensures that pie is drawn as a circle
#plt.axis('equal')

# Add title
plt.title('Pie Chart Example')

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



Start coding or generate with AI.

```
import matplotlib.pyplot as plt

# Data
x = [1, 2, 3, 4, 5]
y = [5, 7, 8, 2, 4]
sizes = [20, 50, 80, 200, 100]  # Size of each point

# Create scatter plot
plt.scatter(x, y, s=sizes)

# Add labels and title
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Scatter Plot Example')

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




```
import matplotlib.pyplot as plt

# Data
x1 = [1, 2, 3, 4, 5]
y1 = [5, 7, 8, 2, 4]
sizes1 = [20, 50, 80, 200, 100]

x2 = [2, 3, 4, 5, 6]
y2 = [8, 6, 3, 5, 7]
sizes2 = [30, 60, 90, 220, 110]

# Create scatter plot
plt.scatter(x1, y1,  color='blue', marker='o', label='Group 1')
plt.scatter(x2, y2, color='red', marker='s', label='Group 2')

# Add labels and title
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Scatter Plot Example')

# Add legend
plt.legend()

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