```
import plotly.express as px
import plotly.graph_objects as go
import pandas as pd
import numpy as np
import networkx as nx
from sklearn.linear_model import LinearRegression
from datetime import datetime, timedelta
```

Question 1

```
data = pd.DataFrame({
    'Company': ['Apple', 'Microsoft', 'Amazon', 'Google', 'Facebook'],
    'Revenue_2022': [394, 198, 513, 280, 117],
    'Revenue_2023': [420, 215, 540, 310, 130]
})
fig1 = px.bar(data, x='Company', y=['Revenue_2022', 'Revenue_2023'],
              barmode='group', title="Company Revenue (2022 vs 2023)")
fig1.show()
```

Question 2

```
data = pd.DataFrame({
    'Advertising Budget': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100],
    'Sales Revenue': [15, 25, 40, 50, 65, 80, 85, 100, 120, 140]
})
fig2 = px.scatter(data, x='Advertising Budget', y='Sales Revenue',
                  color='Sales Revenue', size='Sales Revenue',
                  title="Advertising Budget vs Sales Revenue")
fig2.show()
```

Question 3

```
data = pd.DataFrame({
    'Brand': ['Apple', 'Samsung', 'Xiaomi', 'Oppo', 'Vivo'],
    'Market Share': [30, 28, 17, 12, 13]
})
fig3 = px.pie(data, names='Brand', values='Market Share',
              title="Smartphone Market Share", hover_data=['Market
Share'])
fig3.show()
```

Question 4

```
data = pd.DataFrame({
    'Job Sector': ['IT', 'Finance', 'Healthcare', 'Education',
'Retail'] * 5,
    'Salary': [75000, 85000, 62000, 48000, 40000, 77000, 90000, 65000,
50000, 42000,
                  78000, 87000, 67000, 52000, 43000, 80000, 89000, 69000,
54000, 45000,
                  82000, 91000, 71000, 56000, 47000]
})
fig4 = px.box(data, x='Job Sector', y='Salary', title="Salary
Distribution by Job Sector")
fig4.show()
```

Question 5

```
data = pd.DataFrame({
    'GDP': [19352, 47585, 36867, 30334, 8644, 3846, 43442, 30454,
35695],
    'Inflation': [0.59, 4.86, 4.24, 1.45, 1.31, 1.86, 2.86, 2.44,
1.81],
    'Unemployment': [10.3, 4.67, 6.50, 7.39, 8.47, 5.39, 9.17, 10.1,
3.55],
    'Interest Rate': [6.46, 2.53, 1.58, 9.53, 9.69, 3.74, 1.87, 7.15,
4.96]
})
fig5 = px.imshow(data.corr(), text_auto=True, title="Correlation
Heatmap")
fig5.show()
```

Question 6

```
data = pd.DataFrame({
    'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun'],
    'Product A Sales': [500, 600, 700, 800, 750, 780],
    'Product B Sales': [400, 450, 470, 490, 520, 550],
    'Profit': [50, 80, 100, 120, 110, 130]
})
fig6 = go.Figure()
fig6.add_trace(go.Bar(x=data['Month'], y=data['Product A Sales'],
name="Product A Sales"))
fig6.add_trace(go.Scatter(x=data['Month'], y=data['Profit'],
mode='lines+markers', name="Profit"))
fig6.update_layout(title="Sales & Profit Analysis")
fig6.show()
```

Question 7

```
data = pd.DataFrame({
    'Country': ['USA', 'China', 'India', 'Germany', 'Brazil'] * 3,
    'Year': [2000]*5 + [2010]*5 + [2020]*5,
    'GDP': [10, 5, 2, 3, 1, 15, 9, 5, 4, 2, 22, 14, 7, 5, 3],
    'Population': [280, 1260, 1000, 83, 175, 310, 1350, 1200, 82, 190,
331, 1440, 1380, 80, 210],
    'Life Expectancy': [77, 71, 65, 80, 68, 79, 74, 69, 82, 72, 81,
76, 72, 83, 75]
})
fig7 = px.scatter(data, x="Population", y="Life Expectancy",
animation_frame="Year",
                  size="GDP", color="Country", title="Economic Growth
Animation")
fig7.show()
```

Question 8

```
edges = pd.DataFrame({
    'User A': ['Alice', 'Alice', 'Bob', 'Charlie', 'David', 'Eve',
'Frank', 'Grace', 'Hannah', 'Ivan'],
    'User B': ['Bob', 'Charlie', 'David', 'Eve', 'Frank', 'Grace',
'Hannah', 'Ivan', 'Alice', 'Bob']
})
G = nx.from_pandas_edgelist(edges, 'User A', 'User B')
pos = nx.spring_layout(G)
fig8 = go.Figure()
for edge in G.edges():
    fig8.add_trace(go.Scatter(x=[pos[edge[0]][0], pos[edge[1]][0]],
                              y=[pos[edge[0]][1], pos[edge[1]][1]],
                              mode='lines'))
fig8.show()
```

Question 9

```
s = """Date Stock_Price
01-01-2024 102.4836
02-01-2024 100.3188
03-01-2024 105.2586
04-01-2024 110.6455
05-01-2024 102.8696
06-01-2024 103.8798
07-01-2024 113.9567
08-01-2024 110.9079
09-01-2024 105.7334
10-01-2024 111.8037
11-01-2024 107.7839
12-01-2024 108.7825
13-01-2024 113.331
14-01-2024 103.5649
15-01-2024 105.5168
16-01-2024 112.3401
17-01-2024 111.0975
18-01-2024 118.743
19-01-2024 113.6417
20-01-2024 112.1304
```

```
21-01-2024 127.5303
22-01-2024 120.0832
23-01-2024 122.5599
24-01-2024 116.1086
25-01-2024 121.5205
26-01-2024 125.8071
27-01-2024 120.5077
28-01-2024 129.1512
29-01-2024 125.2796
30-01-2024 127.8345
31-01-2024 127.2945
01-02-2024 140.5745
02-02-2024 132.2557
03-02-2024 128.0448
04-02-2024 138.4562
05-02-2024 129.2493
06-02-2024 137.408
07-02-2024 127.5754
08-02-2024 131.7429
09-02-2024 140.3782
10-02-2024 144.0964
11-02-2024 142.271
12-02-2024 141.846
13-02-2024 141.9288
14-02-2024 137.0518
15-02-2024 141.8553
16-02-2024 144.1615
17-02-2024 152.7604
18-02-2024 150.2029
19-02-2024 140.6797
20-02-2024 152.1255
21-02-2024 149.5897
22-02-2024 149.1406
23-02-2024 156.5937
24-02-2024 159.7005
25-02-2024 160.212
26-02-2024 152.3696
27-02-2024 156.0297
28-02-2024 160.2422
29-02-2024 164.4737
01-03-2024 158.2102
02-03-2024 160.6879
03-03-2024 157.0946
04-03-2024 157.6553
05-03-2024 168.7091
06-03-2024 172.4378
07-03-2024 166.3066
08-03-2024 172.6944
09-03-2024 170.495
```

```
10-03-2024 166.4714
11-03-2024 172.514
12-03-2024 179.4074
13-03-2024 172.5481
14-03-2024 181.5606
15-03-2024 161.6487
16-03-2024 179.8671
17-03-2024 177.2029
18-03-2024 176.2827
19-03-2024 179.2467
20-03-2024 169.8601
21-03-2024 179.7097
22-03-2024 183.6037
23-03-2024 190.2178
24-03-2024 181.247
25-03-2024 180.806
26-03-2024 183.3498
27-03-2024 191.4457
28-03-2024 189.5225
29-03-2024 186.2401
30-03-2024 192.4653
31-03-2024 191.3945
01-04-2024 196.7624
02-04-2024 189.419
03-04-2024 192.3011
04-04-2024 192.989
05-04-2024 188.642
06-04-2024 198.4503
07-04-2024 199.2851
08-04-2024 199.0155
09-04-2024 198.8271"""

l = [i.split(" ") for i in s.split('\n')]
df = pd.DataFrame(l[1:], columns=l[0])
df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%Y')
df['Stock_Price'] = pd.to_numeric(df['Stock_Price'])

fig9 = px.scatter(df, x="Date", y="Stock_Price", trendline="ols",
                 title="Stock Price Prediction with Regression")

fig9.show()
```

Question 10

```
iris = px.data.iris()
fig10 = px.scatter(iris, x="sepal_width", y="sepal_length",
color="species",
                   title="Iris Dataset - Sepal Width vs Sepal Length")
fig10.show()
```