mady-prog /
assignment

Code Issues Pull requests Actions Projects Wiki Security In

main assignment / Assignment 3 (1).ipynb

mady-prog Add files via upload 84cd536 · 1 minute ago

531 lines (531 loc) · 167 KB

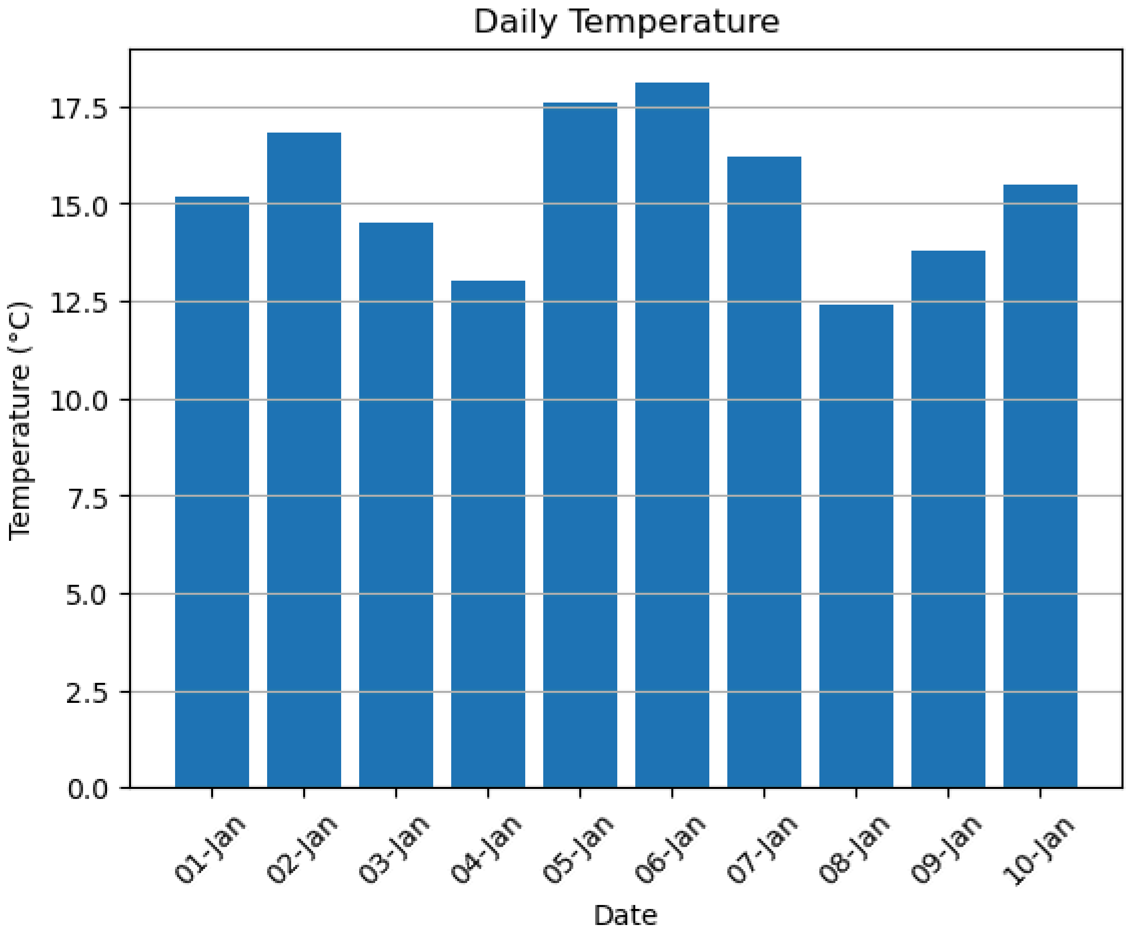
Preview Code Blame Raw

In [32]:

```
"""1. Write a Python script for Weather Data Analysis:
Problem Statement:
You have a CSV file named "weather_data.csv" containing daily weather data for a

Perform the following tasks:
a. Check for missing values and handle them appropriately,
b. Calculate the average temperature for each month,
c. Visualize the monthly average temperature using a bar plot"""
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

d = pd.read_excel("weather.xlsx")
df=pd.DataFrame(d)
print("Original dataframe:\n",df)

#filling NaN values
df1=df.fillna(method="ffill")
print("\nDataframe after filling NaN values:",df1.fillna(method="bfill"))

#Average temperature
average_temperature = df['Temperature (°C)'].mean()
print(f"\nAverage Temperature: {average_temperature:.2f} °C")

# Plotting temperature using a bar chart
df['Date'] = pd.to_datetime(df['Date'], dayfirst=True)
plt.bar(df['Date'].dt.strftime('%d-%b'), df['Temperature (°C)'])
plt.title('Daily Temperature')
plt.xlabel('Date')
plt.ylabel('Temperature (°C)')
plt.grid(axis='y')
plt.xticks(rotation=45)
plt.show()
```

```
Original dataframe:
          Date  Temperature (°C)  Humidity (%)  Wind Speed (km/h)  \
0  01-01-2025              15.2            80               10.5
1  02-01-2025              16.8            75                8.0
2  03-01-2025              14.5            85               12.0
3  04-01-2025              13.0            90               15.2
4  05-01-2025              17.6            70                7.4
5  06-01-2025              18.1            65                6.0
6  07-01-2025              16.2            78                9.1
7  08-01-2025              12.4            88               14.3
8  09-01-2025              13.8            83               11.5
9  10-01-2025              15.5            79               10.0

   Rainfall (mm)
0            NaN
1            NaN
2            1.2
3            5.4
4            NaN
5            NaN
6            0.5
7            2.1
8            NaN
```

```
9            NaN

Dataframe after filling NaN values:          Date  Temperature (°C)  Humidity (%)  
Wind Speed (km/h)  \
0  01-01-2025              15.2            80               10.5   
1  02-01-2025              16.8            75                8.0   
2  03-01-2025              14.5            85               12.0   
3  04-01-2025              13.0            90               15.2   
4  05-01-2025              17.6            70                7.4   
5  06-01-2025              18.1            65                6.0   
6  07-01-2025              16.2            78                9.1   
7  08-01-2025              12.4            88               14.3   
8  09-01-2025              13.8            83               11.5   
9  10-01-2025              15.5            79               10.0   

   Rainfall (mm)  
0            1.2  
1            1.2  
2            1.2  
3            5.4  
4            5.4  
5            5.4  
6            0.5  
7            2.1  
8            2.1  
9            2.1  

Average Temperature: 15.31 °C
```



In [48]:

```
"""2. Write a Python script using pandas for Student Performance Analysis:
```

```
Problem Statement:
You are given a CSV file named &#39;student_scores.csv&#39; containing scores of
various subjects.
Perform the following tasks:
a. Display the summary statistics for each subject,
b. Calculate the average score for each student,
c. Identify students who scored below 60 in more than two subjects."""
import pandas as pd
df = pd.read_excel("scores.xlsx")
print(df)
# a. Summary statistics for each subject
print("Summary Statistics:")
print(df[["Maths", "Physics", "Chemistry"]].describe())

# b. Average score for each student
df["Average Score"] = df[["Maths", "Physics", "Chemistry"]].mean(axis=1)
print("\nStudents with Average Score:")
print(df[["Student ID", "Name", "Average Score"]])

# c. Identify students who scored below 60 in more than two subjects.
below_80 = (df[["Maths", "Physics", "Chemistry"]] < 80).sum(axis=1)
students_below_80 = df[below_80 > 2]
print("\nStudents who scored below 80 in more than two subjects:")
print(students_below_80[["Student ID", "Name", "Maths", "Physics", "Chemistry"]]
```

```
   Student ID         Name  Maths  Physics  Chemistry
0         101  Alice Brown     85       88         87
1         102   Ben Carter     78       75         80
2         103  Clara Davis     92       94         90
3         104  David Evans     65       68         66
4         105   Eva Foster     74       78         76
5         106  Frank Green     60       55         58
6         107   Grace Hall     88       85         86
7         108   Henry Ives     70       72         74
8         109   Isla Jones     95       96         94
9         110    Jack King     82       80         78
Summary Statistics:
           Maths    Physics  Chemistry
count  10.000000  10.000000  10.000000
mean   78.900000  79.100000  78.900000
std    11.618472  12.449453  11.080012
min    60.000000  55.000000  58.000000
25%    71.000000  72.750000  74.500000
50%    80.000000  79.000000  79.000000
75%    87.250000  87.250000  86.750000
max    95.000000  96.000000  94.000000

Students with Average Score:
   Student ID         Name  Average Score
0         101  Alice Brown      86.666667
1         102   Ben Carter      77.666667
2         103  Clara Davis      92.000000
3         104  David Evans      66.333333
4         105   Eva Foster      76.000000
5         106  Frank Green      57.666667
6         107   Grace Hall      86.333333
7         108   Henry Ives      72.000000
8         109   Isla Jones      95.000000
9         110    Jack King      80.000000
```

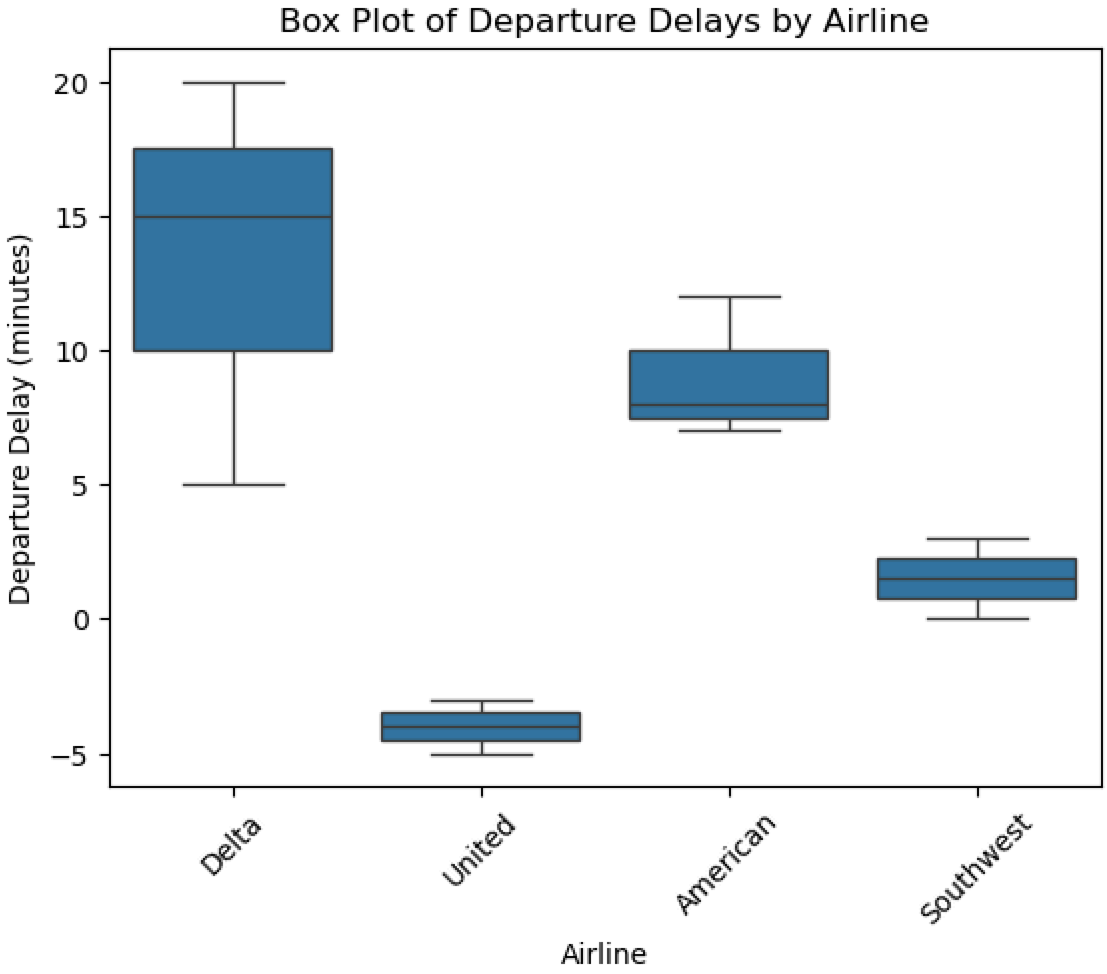
```
Students who scored below 80 in more than two subjects:
   Student ID         Name  Maths  Physics  Chemistry
3         104  David Evans     65       68         66
4         105    Eva Foster     74       78         76
5         106  Frank Green     60       55         58
7         108   Henry Ives     70       72         74
```

In [25]:

```
"""3. Write a Python script for Flight Data Analysis:
Problem Statement:
You have a CSV file named &#39;flight_data.csv&#39; containing flight informatio
destination, and departure time.
Perform the following tasks:

a. Display the summary statistics for departure delays,
b. Calculate the average delay for each airline,
c. Identify the most common departure and arrival destinations,
d. Visualize the distribution of departure delays using a box plot."""
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_excel('flight_data.xlsx')
print("Original DataFrame:\n",df)

# a. Summary statistics for departure delays ---
print("\nSummary Statistics for Departure Delays:")
print(df['DepartureDelay'].describe())

# b. Average delay for each airline ---
print("\nAverage Departure Delay per Airline:")
average_delay = df.groupby('Airline')['DepartureDelay'].mean()
print(average_delay)

# c. Most common departure and arrival destinations ---
print("\nMost Common Origin Airport:", df['Origin'].mode()[0])
print("Most Common Destination Airport:", df['Destination'].mode()[0])

# d. Box plot for distribution of departure delays ---
sns.boxplot(x='Airline', y='DepartureDelay', data=df)
plt.title('Box Plot of Departure Delays by Airline')
plt.ylabel('Departure Delay (minutes)')
plt.xticks(rotation=45)
plt.show()
```

```
Original DataFrame:
      Airline Origin Destination  DepartureDelay
0      Delta    ATL         LAX               5
1     United    ORD         JFK              -3
2   American    DFW         MIA              12
3  Southwest    LAX         ORD               0
4      Delta    ATL         SFO              20
5     United    SFO         ATL              -5
6   American    DFW         LAX               8
7  Southwest    LAX         MIA               3
8      Delta    JFK         ORD              15
9   American    MIA         SFO               7
```

```
Summary Statistics for Departure Delays:
count    10.000000
mean      6.200000
std       7.927449
min      -5.000000
25%       0.750000
50%       6.000000
75%      11.000000
max      20.000000
Name: DepartureDelay, dtype: float64

Average Departure Delay per Airline:
Airline
American      9.000000
Delta        13.333333
Southwest     1.500000
United       -4.000000
Name: DepartureDelay, dtype: float64

Most Common Origin Airport: ATL
Most Common Destination Airport: LAX
```



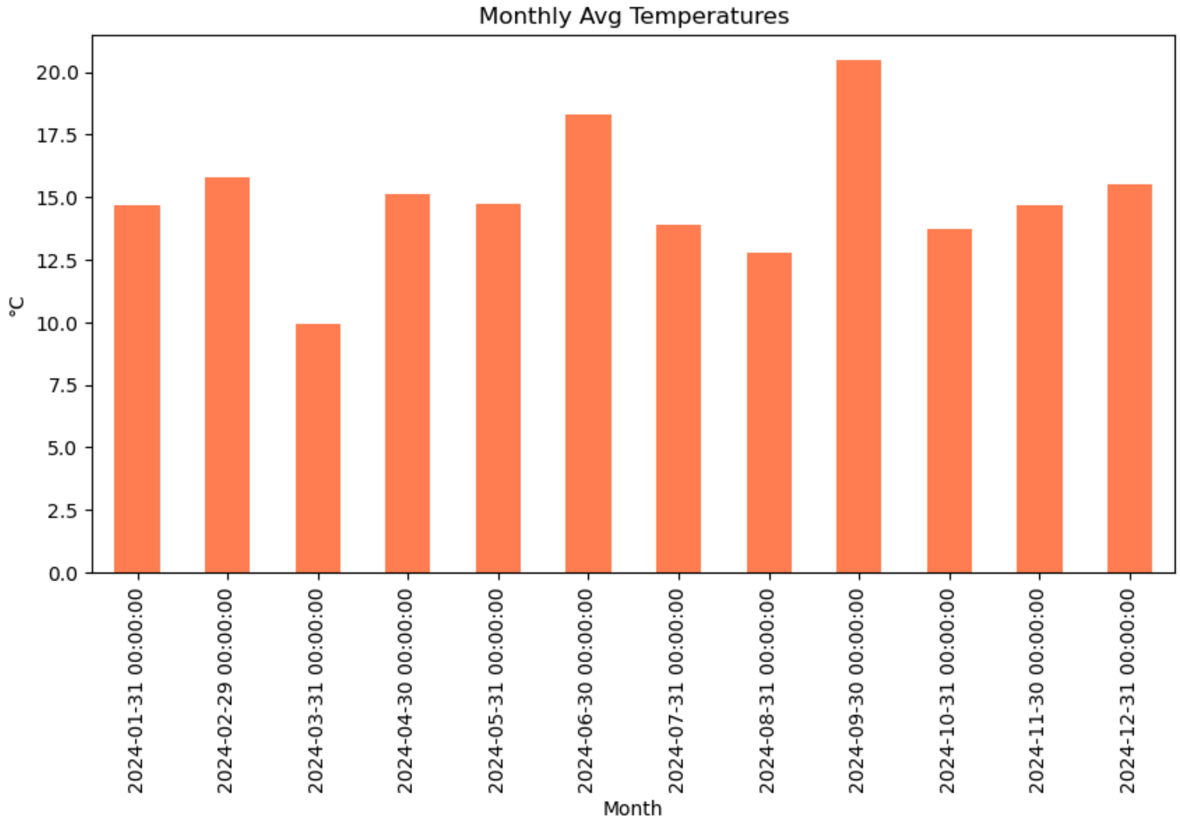
In [9]:

```python
"""4. Write a program to calculate Euclidean distance for two points in 2D space
for two matrix using the mathematical formulas."""
import numpy as np
point1 = np.array([3, 4])
point2 = np.array([7, 1])
distance = np.linalg.norm(point2 - point1)
print(f"Euclidean Distance between {point1} and {point2}: {distance:.2f}")
print(f"Euclidean Distance between {point1} and {point2} = {distance:.2f}")
A = np.array([[1, 2],[3, 4]])
B = np.array([[5, 6],[7, 8]])
dot_result = np.dot(A, B)
print("\nDot Product of the matrices A and B:")
print(dot_result)
```

```
Euclidean Distance between [3 4] and [7 1] = 5.00

Dot Product of the matrices A and B:
[[19 22]
 [43 50]]
```

In [53]:

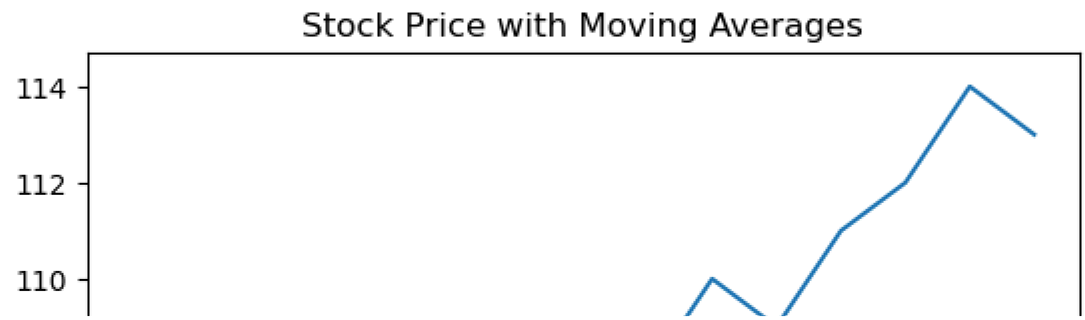
```
"""5. Write a program to work with time series data using NumPy and pandas
Problem Statement: A time series dataset representing daily temperatures over a
You will perform some basic analysis and visualization using NumPy, pandas, and
matplotlib."""
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
dates = pd.date_range('2024-01-01', periods=365)
temps = np.random.uniform(-10, 40, size=365)
df = pd.DataFrame({'Temperature': temps}, index=dates)
monthly_avg = df.resample('ME').mean()
monthly_avg.plot(kind='bar', color='coral', figsize=(10, 5), legend=False)
plt.title("Monthly Avg Temperatures")
plt.ylabel("°C")
plt.xlabel("Month")
plt.show()
```

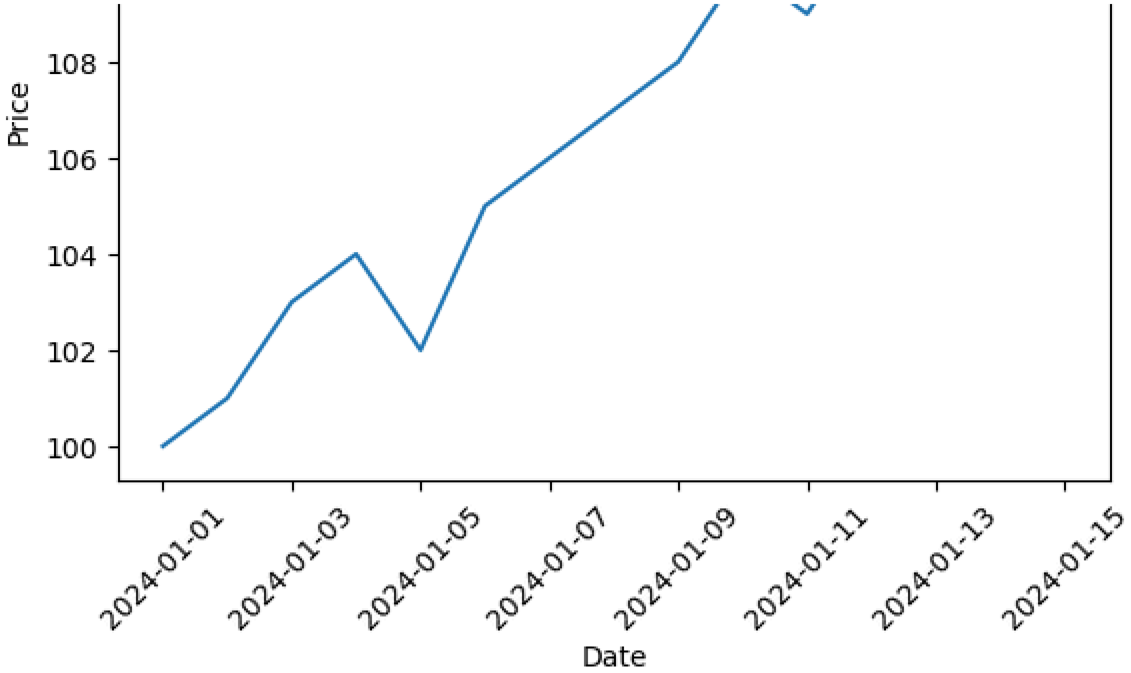


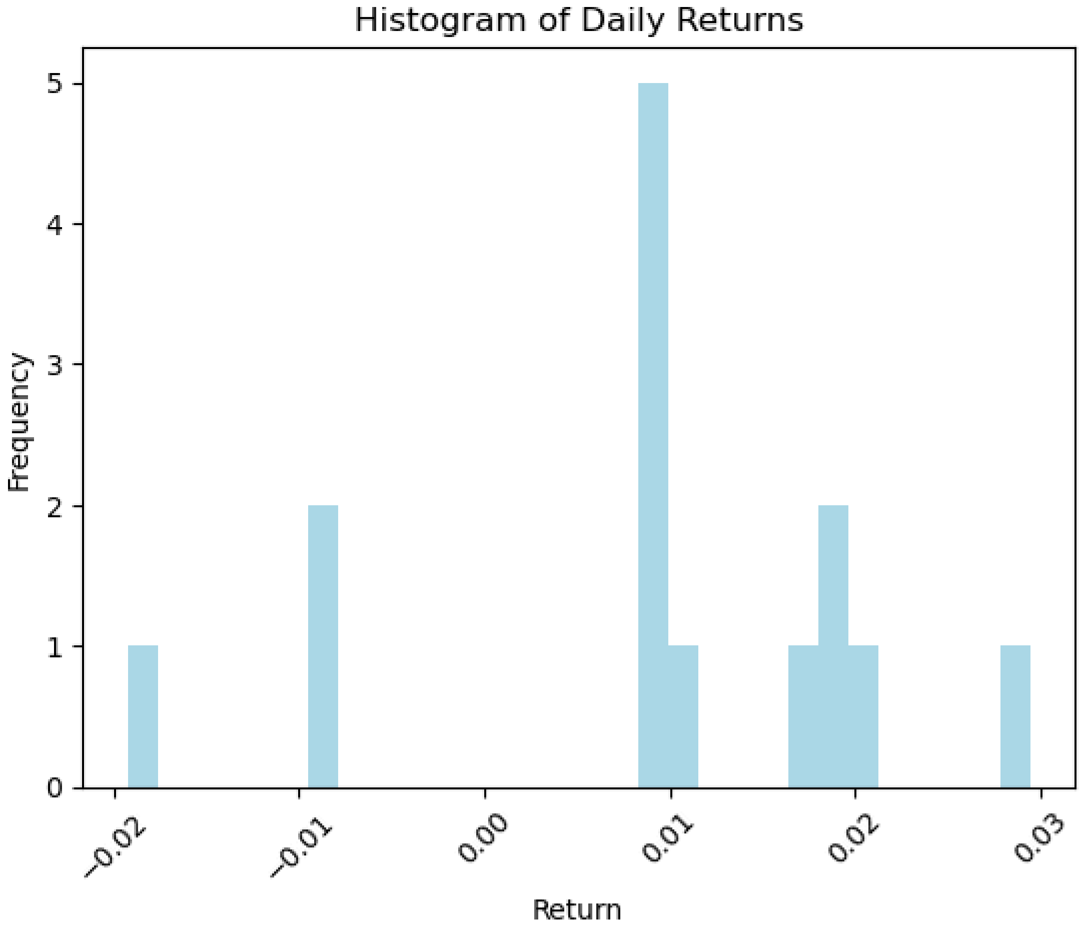
In [77]:

```
In [77]:  """6. Write a program to simulate and analyze daily stock prices over one year a
           the stock prices, calculate some basic metrics, and visualize the data with addi
           features like moving averages and visualize the same."""
           import numpy as np
           import pandas as pd
           import matplotlib.pyplot as plt
           df = pd.read_excel("stock.xlsx")
           print("Original DataFrame:\n",df)
           df['Date'] = pd.to_datetime(df['Date'])
           # 1. Basic statistics
           print("Average Price:", df['Price'].mean())
           print("Standard Deviation:", df['Price'].std())

           # 2. Plot stock price and moving averages
           plt.plot(df['Date'], df['Price'], label='Price')
           plt.title("Stock Price with Moving Averages")
           plt.xlabel("Date")
           plt.ylabel("Price")
           plt.xticks(rotation=45)
           plt.show()

           # 3. Plot histogram of daily returns
           df['DailyReturn'] = df['Price'].pct_change()#pct_change=percentage change
           plt.hist(df['DailyReturn'].dropna(), bins=30, color='lightblue')
           plt.title("Histogram of Daily Returns")
           plt.xlabel("Return")
           plt.ylabel("Frequency")
           plt.xticks(rotation=45)
           plt.show()
```

```
Original DataFrame:
          Date  Price
0  2024-01-01    100
1  2024-01-02    101
2  2024-01-03    103
3  2024-01-04    104
4  2024-01-05    102
5  2024-01-06    105
6  2024-01-07    106
7  2024-01-08    107
8  2024-01-09    108
9  2024-01-10    110
10 2024-01-11    109
11 2024-01-12    111
12 2024-01-13    112
13 2024-01-14    114
14 2024-01-15    113
Average Price: 107.0
Standard Deviation: 4.47213595499958
```







In [83]:

```
"""7. Write a simple Python program using pandas that creates a DataFrame, perfo
following basic operations, and prints the result.
a. Creates two lists: data containing fruit names and prices containing their
corresponding prices. Zips these lists together and creates a DataFrame named
fruits_df with columns named &quot;Fruit&quot; and &quot;Price&quot;.
b. Uses info() to get information about the DataFrame, including data types and
number of entries.
c. Prints the entire DataFrame using to_string().
```

```
d. Calculates descriptive statistics (mean, standard deviation, etc.) for the &q
column and prints the results"""
import pandas as pd
fruit_names = ['Apple', 'Banana', 'Cherry', 'Date', 'Elderberry']
prices = [1.2, 0.5, 3.0, 2.5, 5.0]
fruits_df = pd.DataFrame(list(zip(fruit_names, prices)), columns=['Fruit', 'Pric
print("DataFrame Information:")
fruits_df.info()
print("\nEntire DataFrame:")
print(fruits_df.to_string(index=False))
print("\nDescriptive Statistics for 'Price' column:")
print(fruits_df['Price'].describe())
```

```
DataFrame Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Fruit   5 non-null      object 
 1   Price   5 non-null      float64
dtypes: float64(1), object(1)
memory usage: 212.0+ bytes

Entire DataFrame:
     Fruit  Price
     Apple    1.2
    Banana    0.5
    Cherry    3.0
      Date    2.5
Elderberry    5.0

Descriptive Statistics for 'Price' column:
count    5.00000
mean     2.44000
std      1.74442
min      0.50000
25%      1.20000
50%      2.50000
75%      3.00000
max      5.00000
Name: Price, dtype: float64
```