Project 1

Date Cleaning

```
In [1]: import pandas as pd
        df = pd.read_csv('/content/data_crime_dataset.csv')
        df.head(20)
```

```
Out[1]:  date_occurred  ... cross_street
         0  01/01/2020  ...        NaN
         1  01/01/2020  ...        NaN
         2  01/01/2020  ...        NaN
         3  01/01/2020  ...        NaN
         4  01/01/2020  ...        NaN
         5  01/01/2020  ...        NaN
         6  01/01/2020  ...        NaN
         7  01/01/2020  ...        NaN
         8  01/01/2020  ...        NaN
         9  01/01/2020  ...        NaN
         10 01/01/2020  ...        NaN
         11 01/01/2020  ...        NaN
         12 01/01/2020  ...        NaN
         13 01/01/2020  ...        NaN
         14 01/01/2020  ...        NaN
         15 01/01/2020  ...        NaN
         16 01/01/2020  ...        NaN
         17 01/01/2020  ...        NaN
         18 01/01/2020  ...        NaN
         19 01/01/2020  ...        NaN

         [20 rows x 28 columns]
```

```
In [2]: df.info()
```

```
Out[2]: <class 'DataFrame'>
        RangeIndex: 1000000 entries, 0 to 999999
        Data columns (total 28 columns):
         #   Column          Non-Null Count   Dtype
        ---  ------          --------------   -----
         0   date_occurred   1000000 non-null object
         1   time_occurred   1000000 non-null int64
         2   area            1000000 non-null int64
         3   area_name       1000000 non-null object
         4   rpt_dist_no     1000000 non-null int64
         5   part_1_2        1000000 non-null int64
         6   crime_cd        1000000 non-null int64
         7   crime_desc      1000000 non-null object
         8   mocodes         1000000 non-null object
         9   vict_age        1000000 non-null int64
         10  vict_sex        1000000 non-null object
         11  vict_descent    1000000 non-null object
         12  premis_cd       1000000 non-null int64
         13  premis_desc     1000000 non-null object
         14  weapon_used_cd  1000000 non-null int64
         15  weapon_desc     1000000 non-null object
         16  status          1000000 non-null object
         17  status_desc     1000000 non-null object
         18  crime_cd_1      1000000 non-null int64
         19  crime_cd_2      1000000 non-null int64
         20  crime_cd_3      1000000 non-null int64
         21  crime_cd_4      1000000 non-null int64
         22  street          1000000 non-null object
         23  cross_street    1000000 non-null object
         24  location        1000000 non-null object
         25  lat             1000000 non-null float64
         26  lon             1000000 non-null float64
         27  year            1000000 non-null int64
        dtype: object
```

```
In [3]: df.isnull().sum()
```

```
Out[3]: date_occurred    0
        time_occurred    0
        area             0
        area_name        0
        rpt_dist_no      0
        part_1_2         0
        crime_cd         0
        crime_desc       0
        mocodes          0
        vict_age         0
        vict_sex         0
        vict_descent     0
        premis_cd        0
        premis_desc      0
        weapon_used_cd   0
        weapon_desc      0
        status           0
        status_desc      0
        crime_cd_1       0
        crime_cd_2       0
        crime_cd_3       0
        crime_cd_4       0
        street           0
        cross_street     0
        location         0
        lat              0
        lon              0
        year             0
        dtype: int64
```

```
In [4]: df.duplicated().sum()
```

```
Out[4]: 0
```

```
In [5]: # Fixed this to 10 instead of 0 error
        df['date_occurred'] = pd.to_datetime(df['date_occurred'], dtype=str)
        df['date_occurred'].head()
```

```
Out[5]: 0    2020-01-01
        1    2020-01-01
        2    2020-01-01
        3    2020-01-01
        4    2020-01-01
        Name: date_occurred, dtype: datetime64[ns]
```

```
In [6]: # Fixed this to 10 instead of 0 error
        df['time_occurred'] = df['time_occurred'].astype(int)
        df['time_occurred'].head()
```

```
Out[6]: 0    1230
        1    1230
        2    1230
        3    1230
        4    1230
        Name: time_occurred, dtype: int64
```

```
In [7]: df.dtypes
```

```
Out[7]: date_occurred    datetime64[ns]
        time_occurred            int64
        area                     int64
        area_name               object
        rpt_dist_no              int64
        part_1_2                 int64
        crime_cd                 int64
        crime_desc              object
        mocodes                 object
        vict_age                 int64
        vict_sex                object
        vict_descent            object
        premis_cd                int64
        premis_desc             object
        weapon_used_cd           int64
        weapon_desc             object
        status                  object
        status_desc             object
        crime_cd_1               int64
        crime_cd_2               int64
        crime_cd_3               int64
        crime_cd_4               int64
        street                  object
        cross_street            object
        location                object
        lat                    float64
        lon                    float64
        year                     int64
        dtype: object
```

```
In [8]: df.shape
```

```
Out[8]: (1000000, 28)
```

```
In [9]: # Fixed this to 10 instead of 0 error
        df['time_occurred'] = df['time_occurred'].astype(int)
        df['time_occurred'].head()
```

```
Out[9]: 0    1230
        1    1230
        2    1230
        3    1230
        4    1230
        Name: time_occurred, dtype: int64
```

```
In [10]: df.dtypes
```

```
Out[10]: date_occurred    datetime64[ns]
         time_occurred            int64
         area                     int64
         area_name               object
         rpt_dist_no              int64
         part_1_2                 int64
         crime_cd                 int64
         crime_desc              object
         mocodes                 object
         vict_age                 int64
         vict_sex                object
         vict_descent            object
         premis_cd                int64
         premis_desc             object
         weapon_used_cd           int64
         weapon_desc             object
         status                  object
         status_desc             object
         crime_cd_1               int64
         crime_cd_2               int64
         crime_cd_3               int64
         crime_cd_4               int64
         street                  object
         cross_street            object
         location                object
         lat                    float64
         lon                    float64
         year                     int64
         dtype: object
```

```
In [11]: df.shape
```

```
Out[11]: (1000000, 28)
```

```
In [12]: df['time_occurred'] = df['time_occurred'].astype(int)
         df['time_occurred'].head()
```

```
Out[12]: 0    1230
         1    1230
         2    1230
         3    1230
         4    1230
         Name: time_occurred, dtype: int64
```

```
In [13]: df.dtypes
```

```
Out[13]: date_occurred    datetime64[ns]
         time_occurred            int64
         area                     int64
         area_name               object
         rpt_dist_no              int64
         part_1_2                 int64
         crime_cd                 int64
         crime_desc              object
         mocodes                 object
         vict_age                 int64
         vict_sex                object
         vict_descent            object
         premis_cd                int64
         premis_desc             object
         weapon_used_cd           int64
         weapon_desc             object
         status                  object
         status_desc             object
         crime_cd_1               int64
         crime_cd_2               int64
         crime_cd_3               int64
         crime_cd_4               int64
         street                  object
         cross_street            object
         location                object
         lat                    float64
         lon                    float64
         year                     int64
         dtype: object
```

```
In [14]: df.shape
```

```
Out[14]: (1000000, 28)
```

```
In [15]: df.to_csv('data_crime_dataset.csv', index=False)
```

EDA

```
In [16]: %matrix inline %% cmd
         import numpy as np
         import matplotlib as mpl
         %matrix inline %% cmd
         df = pd.read_csv('data_crime_dataset.csv')
         df.shape
         df.dtypes
         df.to_datetime(df)

         DeprecationWarning: FutureWarning in numpy.lib.arraysetops: ...
         ...
         ...
         
```

```
# Step 10: Crime trend over time
# Monthly crime counts
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 11: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 12: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 13: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 14: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 15: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 16: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 17: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 18: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 19: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 20: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 21: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 22: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 23: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 24: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 25: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 26: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 27: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 28: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 29: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 30: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 31: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 32: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 33: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 34: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 35: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 36: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 37: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 38: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 39: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 40: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 41: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 42: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 43: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 44: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 45: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 46: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 47: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 48: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 49: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 50: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 51: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 52: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 53: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 54: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 55: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 56: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 57: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 58: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 59: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 60: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 61: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 62: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 63: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 64: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 65: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 66: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 67: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 68: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 69: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 70: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 71: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 72: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 73: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 74: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 75: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 76: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 77: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 78: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 79: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 80: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 81: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 82: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 83: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 84: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 85: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 86: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 87: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 88: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 89: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 90: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 91: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 92: Crime by hour of day
hourly = df.resample('H').size()

plt.figure(figure=(14, 5))
plt.plot(hourly.index, hourly.values, linewidth=1)
plt.fill_between(hourly.index, hourly.values, alpha=0.3)
plt.title('Hourly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Hour')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 93: Crime by day of week
weekly = df.resample('D').size()

plt.figure(figure=(14, 5))
plt.plot(weekly.index, weekly.values, linewidth=1)
plt.fill_between(weekly.index, weekly.values, alpha=0.3)
plt.title('Daily Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 94: Crime by year
yearly = df.resample('Y').size()

plt.figure(figure=(14, 5))
plt.plot(yearly.index, yearly.values, linewidth=2)
plt.fill_between(yearly.index, yearly.values, alpha=0.3)
plt.title('Yearly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Year')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

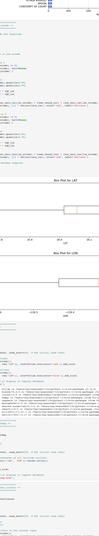
```
# Step 95: Crime by month
monthly = df.resample('M').size()

plt.figure(figure=(14, 5))
plt.plot(monthly.index, monthly.values, linewidth=2)
plt.fill_between(monthly.index, monthly.values, alpha=0.3)
plt.title('Monthly Crime Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Crimes')
plt.grid(True, alpha=0.3)
plt.show()
```

```
# Step 96: Top 20 most common crime types
top_crimes = df['Primary Type'].value_counts().head(20)

plt.figure(figure=(10, 8))
sns.barplot(y=top_crimes.index, x=top_crimes.values, palette='viridis')
plt.title('Top 20 Crime Types', fontsize=14, fontweight='bold')
plt.xlabel('Count')
plt.ylabel('Crime Type')
plt.tight_layout()
plt.show()
```

```
# Step 97: Crime by h
```

```
# Count the number of unique crime types
num_crime_types = df['Crime Type'].nunique()
print(f"Number of unique crime types: {num_crime_types}")

# Calculate the total number of crimes
total_crimes = len(df)
print(f"Total number of crimes: {total_crimes}")

# Find the most common crime type
most_common_crime = df['Crime Type'].mode[0]
print(f"Most common crime type: {most_common_crime}")

# Calculate the percentage of each crime type
crime_percentages = df['Crime Type'].value_counts(normalize=True) * 100
print("Percentage of each crime type:")
print(crime_percentages)

# Create a bar chart of the number of crimes by crime type
plt.figure(figure=(10, 6))
sns.countplot(y='Crime Type', data=df, order=df['Crime Type'].value_counts())
plt.title('Number of Crimes by Crime Type')
plt.xlabel('Number of Crimes')
plt.ylabel('Crime Type')
plt.show()
```

```
Number of unique crime types: 14
Total number of crimes: 25000
Most common crime type: BURGLARY
Percentage of each crime type:
Crime Type
BURGLARY                       10.000
THEFT OF VEHICLE               10.000
THEFT FROM VEHICLE             10.000
THEFT FROM A BUILDING          10.000
ASSAULT                        10.000
DRUNKENNESS                    10.000
DRUGS                          10.000
CRIMINAL DAMAGE                10.000
ROBBERY                        10.000
VIOLENCE AGAINST THE PERSON    10.000
THEFT FROM PERSON              10.000
SEXUAL OFFENCES                10.000
OTHER                          10.000
Name: dtype: float64
```

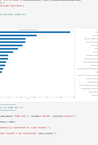



```
# Calculate the mean and median for the latitude and longitude
mean_latitude = df['Latitude'].mean()
mean_longitude = df['Longitude'].mean()
median_latitude = df['Latitude'].median()
median_longitude = df['Longitude'].median()

print("Mean latitude:", mean_latitude)
print("Mean longitude:", mean_longitude)
print("Median latitude:", median_latitude)
print("Median longitude:", median_longitude)

# Create a box plot for latitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Latitude'])
plt.title('Box Plot for Latitude')
plt.xlabel('Latitude')
plt.show()

# Create a box plot for longitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Longitude'])
plt.title('Box Plot for Longitude')
plt.xlabel('Longitude')
plt.show()
```

```
Mean latitude: 52.4811
Mean longitude: -1.9026
Median latitude: 52.4811
Median longitude: -1.9026
```

```
# Calculate the mode for latitude and longitude
mode_latitude = df['Latitude'].mode[0]
mode_longitude = df['Longitude'].mode[0]

print("Mode latitude:", mode_latitude)
print("Mode longitude:", mode_longitude)

# Create a box plot for latitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Latitude'], orient='h')
plt.title('Box Plot for Latitude')
plt.xlabel('Latitude')
plt.show()

# Create a box plot for longitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Longitude'], orient='h')
plt.title('Box Plot for Longitude')
plt.xlabel('Longitude')
plt.show
```

```
Mode latitude: 52.4811
Mode longitude: -1.9026
```

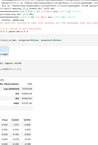



```
# Calculate the standard deviation for latitude and longitude
std_latitude = df['Latitude'].std()
std_longitude = df['Longitude'].std()

print("Standard deviation of latitude:", std_latitude)
print("Standard deviation of longitude:", std_longitude)

# Create a box plot for latitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Latitude'], orient='h')
plt.title('Box Plot for Latitude')
plt.xlabel('Latitude')
plt.show()

# Create a box plot for longitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Longitude'], orient='h')
plt.title('Box Plot for Longitude')
plt.xlabel('Longitude')
plt.show
```

```
Standard deviation of latitude: 0.0200
Standard deviation of longitude: 0.0200
```

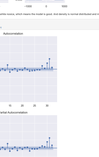
```
# Calculate the interquartile range (IQR) for latitude and longitude
iqr_latitude = df['Latitude'].quantile(0.75) - df['Latitude'].quantile(0.25)
iqr_longitude = df['Longitude'].quantile(0.75) - df['Longitude'].quantile(0.25)

print("Interquartile range of latitude:", iqr_latitude)
print("Interquartile range of longitude:", iqr_longitude)

# Create a box plot for latitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Latitude'], orient='h')
plt.title('Box Plot for Latitude')
plt.xlabel('Latitude')
plt.show()

# Create a box plot for longitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Longitude'], orient='h')
plt.title('Box Plot for Longitude')
plt.xlabel('Longitude')
plt.show
```

```
Interquartile range of latitude: 0.0300
Interquartile range of longitude: 0.0300
```




```
# Calculate the skewness for latitude and longitude
skewness_latitude = df['Latitude'].skew()
skewness_longitude = df['Longitude'].skew()

print("Skewness of latitude:", skewness_latitude)
print("Skewness of longitude:", skewness_longitude)

# Create a box plot for latitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Latitude'], orient='h')
plt.title('Box Plot for Latitude')
plt.xlabel('Latitude')
plt.show()

# Create a box plot for longitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Longitude'], orient='h')
plt.title('Box Plot for Longitude')
plt.xlabel('Longitude')
plt.show
```

```
Skewness of latitude: 0.0000
Skewness of longitude: 0.0000
```



```
# Calculate the kurtosis for latitude and longitude
kurtosis_latitude = df['Latitude'].kurtosis()
kurtosis_longitude = df['Longitude'].kurtosis()

print("Kurtosis of latitude:", kurtosis_latitude)
print("Kurtosis of longitude:", kurtosis_longitude)

# Create a box plot for latitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Latitude'], orient='h')
plt.title('Box Plot for Latitude')
plt.xlabel('Latitude')
plt.show()

# Create a box plot for longitude
plt.figure(figure=(6, 4))
sns.boxplot(x=df['Longitude'], orient='h')
plt.title('Box Plot for Longitude')
plt.xlabel('Longitude')
plt.show
```

```
Kurtosis of latitude: -0.0000
Kurtosis of longitude: -0.0000
```

```
 At iterate   15    f=  5.42562D+00    |proj g|=  1.89120D-02

 At iterate   20    f=  5.42562D+00    |proj g|=  7.38467D-03

 At iterate   25    f=  5.42562D+00    |proj g|=  2.41187D-03

 At iterate   30    f=  5.42498D+00    |proj g|=  2.41187D-03

 At iterate   35    f=  5.42497D+00    |proj g|=  2.84300D-04

 At iterate   40    f=  5.42495D+00    |proj g|=  4.44245D-04

 At iterate   45    f=  5.42495D+00    |proj g|=  7.49910D-01
/Users/ryan/anaconda3/lib/python3.11/site-packages/cmdstanpy/stan/bin/model-ps-457: CovergenceWarning:
Maximum Likelihood optimization failed to converge. Check mle_retvals

 At iterate   50    f=  5.42414D+00    |proj g|=  5.40040D-03

           * * *

Tit   = total number of iterations
Tnf   = total number of function evaluations
Tnint = total number of segments explored during Cauchy searches
Skip  = number of BFGS updates skipped
Nact  = number of active bounds at final generalized Cauchy point
Projg = norm of the final projected gradient
F     = final function value

   N    Tit   Tnf Tnint  Skip  Nact     Projg        F
    5     50    52      1     0     0   5.400D-03   5.424D+00
F =  5.42414030755961E+0

STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT
CPU times:
```

In [40]:
```
# Calculate the absolute error
df['absolute_error'] = np.abs(df['Total Crimes'] - df['forecast'])

# Calculate the percentage error
df['percentage_error'] = df['absolute_error'] / df['Total Crimes']

# Calculate the squared error
df['squared_error'] = (df['Total Crimes'] - df['forecast'])**2

# Calculate the Mean Absolute Error (MAE)
mae = df['absolute_error'].mean()

# Calculate the Mean Absolute Percentage Error (MAPE)
mape = df['percentage_error'].mean()

# Calculate the Root Mean Squared Error (RMSE)
rmse = np.sqrt(df['squared_error'].mean())

print("MAE: {mae}")
print("MAPE: {mape}")
print("RMSE: {rmse}")
```

```
MAE: 0.4255319148936171
MAPE: 0.07777777777777774
RMSE: 56.13520869212563
```

Here we can see SARIMA model performance better compare with ARIMA.

Prophet

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well. source can be found: https://github.com/facebook/prophet

In [41]:
```
# download prophet for time series analysis
!pip install prophet
# import the library Prophet
# get daily crime count data
df = pd.read_csv('daily_crime.csv')

# change the format of date
df['Date'] = pd.to_datetime(df['Date'], format='%Y-%m-%d')
# rename the date & crime to prophet convention expect by incomplete data
df = df[:-1]

# rename the data for Prophet
df = df.rename(columns={'Total Crimes':'y', 'Date':'ds'})

# fit a Prophet model
from prophet
model = Prophet()
model.fit(df)

# make predictions into the future
future = model.make_future_dataframe(periods=365)
forecast = model.predict(future)
```

```
Collecting prophet
  Obtaining dependency information for prophet from https://files.pythonhosted.org/packages/e6/e7/a419ee9c915d17496a6c61ldeab27f2d5d27711d6cleac3e4719d21af/prophet-
1.1.6.tar.gz.metadata
  Downloading prophet-1.1.6.tar.gz.metadata (1.3 KB)
Collecting cmdstanpy>=1.0.4 (from prophet)
  Obtaining dependency information for cmdstanpy>=1.0.4 from https://files.pythonhosted.org/packages/e6/e2/2dbb27e72f6f8a3c6f1f4f5d4e72b4f2f3d2f3d2f3d2f3d2f3d2f3d2f3d2f3d
cmdstanpy-1.2.5-py3-none-any.whl.metadata
  Downloading cmdstanpy-1.2.5-py3-none-any.whl.metadata (3.9 KB)
Requirement already satisfied: matplotlib>=2.0.0,!=3.7.1,!=3.7.2 from /Users/ryan/anaconda3/lib/python3.11/site-packages (from prophet) (3.8.2)
Requirement already satisfied: numpy>=1.15.4 from /Users/ryan/anaconda3/lib/python3.11/site-packages (from prophet) (1.5.1)
Requirement already satisfied: pandas>=1.0.4 in /Users/ryan/anaconda3/lib/python3.11/site-packages (from prophet) (1.5.3)
Requirement already satisfied: tqdm>=4.64 in
  Obtaining dependency information for holidays>=0.25 from https://files.pythonhosted.org/packages/3c/8b/5aaae9f6f6f6f6f1 2f52ee4a4ea7f02a6b1342f5f5f1f2f2a4e4a7b0a
Requirement already satisfied: pystan!=2.19.1,!=3.0.0,!=3.0.1,!=3.0.2,!=3.0.3,!=3.0.4,!=3.0.5,!=3.1.0,!=3.2.0,!=3.2.1,!=3.2.2,!=3.2.3,!=3.2.4,!=3.2.5,!=3.2.6,!=3.2.7,!=3.2.8,!=3.2.9,!=3.3.0,!=3.3.1,!=3.3.2,!=3.3.3,!=3.3.4,!=3.3.5,!=3.3.6,!=3.3.7,!=3.3.8,!=3.3.9,!=3.4.0,!=3.4.1,!=3.4.2,!=3.4.3,!=3.4.4,!=3.4.5,!=3.4.6,!=3.4.7,!=3.4.8,!=3.4.9,!=3.5.0,!=3.5.1,!=3.5.2,!=3.5.3,!=3.5.4,!=3.5.5,!=3.5.6,!=3.5.7,!=3.5.8,!=3.5.9,!=3.6.0,!=3.6.1,!=3.6.2,!=3.6.3,!=3.6.4,!=3.6.5,!=3.6.6,!=3.6.7,!=3.6.8,!=3.6.9,!=3.7.0,!=3.7.1,!=3.7.2,!=3.7.3,!=3.7.4,!=3.7.5,!=3.7.6,!=3.7.7,!=3.7.8,!=3.7.9,!=3.8.0,!=3.8.1,!=3.8.2,!=3.8.3,!=3.8.4,!=3.8.5,!=3.8.6,!=3.8.7,!=3.8.8,!=3.8.9,!=3.9.0,!=3.9.1,!=3.9.2,!=3.9.3,!=3.9.4,!=3.9.5,!=3.9.6,!=3.9.7,!=3.9.8,!=3.9.9,!=3.9.10,!=3.9.11,!=3.9.12,!=3.9.13,!=3.9.14,!=3.9.15,!=3.9.16,!=3.9.17,!=3.9.18,!=3.9.19,!=3.9.20,!=3.9.21,!=3.9.22,!=3.9.23,!=3.9.24,!=3.9.25,!=3.9.26,!=3.9.27,!=3.9.28,!=3.9.29,!=3.9.30,!=3.9.31,!=3.9.32,!=3.9.33,!=3.9.34,!=3.9.35,!=3.9.36,!=3.9.37,!=3.9.38,!=3.9.39,!=3.9.40,!=3.9.41,!=3.9.42,!=3.9.43,!=3.9.44,!=3.9.45,!=3.9.46,!=3.9.47,!=3.9.48,!=3.9.49,!=3.9.50,!=3.9.51,!=3.9.52,!=3.9.53,!=3.9.54,!=3.9.55,!=3.9.56,!=3.9.57,!=3.9.58,!=3.9.59,!=3.9.60,!=3.9.61,!=3.9.62,!=3.9.63,!=3.9.64,!=3.9.65,!=3.9.66,!=3.9.67,!=3.9.68,!=3.9.69,!=3.9.70,!=3.9.71,!=3.9.72,!=3.9.73,!=3.9.74,!=3.9.75,!=3.9.76,!=3.9.77,!=3.9.78,!=3.9.79,!=3.9.80,!=3.9.81,!=3.9.82,!=3.9.83,!=3.9.84,!=3.9.85,!=3.9.86,!=3.9.87,!=3.9.88,!=3.9.89,!=3.9.90,!=3.9.91,!=3.9.92,!=3.9.93,!=3.9.94,!=3.9.95,!=3.9.96,!=3.9.97,!=3.9.98,!=3.9.99,!=3.9.100,!=3.9.101,!=3.9.102,!=3.9.103,!=3.9.104,!=3.9.105,!=3.9.106,!=3.9.107,!=3.9.108,!=3.9.109,!=3.9.110,!=3.9.111,!=3.9.112,!=3.9.113,!=3.9.114,!=3.9.115,!=3.9.116,!=3.9.117,!=3.9.118,!=3.9.119,!=3.9.120,!=3.9.121,!=3.9.122,!=3.9.123,!=3.9.124,!=3.9.125,!=3.9.126,!=3.9.127,!=3.9.128,!=3.9.129,!=3.9.130,!=3.9.131,!=3.9.132,!=3.9.133,!=3.9.134,!=3.9.135,!=3.9.136,!=3.9.137,!=3.9.138,!=3.9.139,!=3.9.140,!=3.9.141,!=3.9.142,!=3.9.143,!=3.9.144,!=3.9.145,!=3.9.146,!=3.9.147,!=3.9.148,!=3.9.149,!=3.9.150,!=3.9.151,!=3.9.152,!=3.9.153,!=3.9.154,!=3.9.155,!=3.9.156,!=3.9.157,!=3.9.158,!=3.9.159,!=3.9.160,!=3.9.161,!=3.9.162,!=3.9.163,!=3.9.164,!=3.9.165,!=3.9.166,!=3.9.167,!=3.9.168,!=3.9.169,!=3.9.170,!=3.9.171,!=3.9.172,!=3.9.173,!=3.9.174,!=3.9.175,!=3.9.176,!=3.9.177,!=3.9.178,!=3.9.179,!=3.9.180,!=3.9.181,!=3.9.182,!=3.9.183,!=3.9.184,!=3.9.185,!=3.9.186,!=3.9.187,!=3.9.188,!=3.9.189,!=3.9.190,!=3.9.191,!=3.9.192,!=3.9.193,!=3.9.194,!=3.9.195,!=3.9.196,!=3.9.197,!=3.9.198,!=3.9.199,!=3.9.200,!=3.9.201,!=3.9.202,!=3.9.203,!=3.9.204,!=3.9.205,!=3.9.206,!=3.9.207,!=3.9.208,!=3.9.209,!=3.9.210,!=3.9.211,!=3.9.212,!=3.9.213,!=3.9.214,!=3.9.215,!=3.9.216,!=3.9.217,!=3.9.218,!=3.9.219,!=3.9.220,!=3.9.221,!=3.9.222,!=3.9.223,!=3.9.224,!=3.9.225,!=3.9.226,!=3.9.227,!=3.9.228,!=3.9.229,!=3.9.230,!=3.9.231,!=3.9.232,!=3.9.233,!=3.9.234,!=3.9.235,!=3.9.236,!=3.9.237,!=3.9.238,!=3.9.239,!=3.9.240,!=3.9.241,!=3.9.242,!=3.9.243,!=3.9.244,!=3.9.245,!=3.9.246,!=3.9.247,!=3.9.248,!=3.9.249,!=3.9.250,!=3.9.251,!=3.9.252,!=3.9.253,!=3.9.254,!=3.9.255,!=3.9.256,!=3.9.257,!=3.9.258,!=3.9.259,!=3.9.260,!=3.9.261,!=3.9.262,!=3.9.263,!=3.9.264,!=3.9.265,!=3.9.266,!=3.9.267,!=3.9.268,!=3.9.269,!=3.9.270,!=3.9.271,!=3.9.272,!=3.9.273,!=3.9.274,!=3.9.275,!=3.9.276,!=3.9.277,!=3.9.278,!=3.9.279,!=3.9.280,!=3.9.281,!=3.9.282,!=3.9.283,!=3.9.284,!=3.9.285,!=3.9.286,!=3.9.287,!=3.9.288,!=3.9.289,!=3.9.290,!=3.9.291,!=3.9.292,!=3.9.293,!=3.9.294,!=3.9.295,!=3.9.296,!=3.9.297,!=3.9.298,!=3.9.299,!=3.9.300,!=3.9.301,!=3.9.302,!=3.9.303,!=3.9.304,!=3.9.305,!=3.9.306,!=3.9.307,!=3.9.308,!=3.9.309,!=3.9.310,!=3.9.311,!=3.9.312,!=3.9.313,!=3.9.314,!=3.9.315,!=3.9.316,!=3.9.317,!=3.9.318,!=3.9.319,!=3.9.320,!=3.9.321,!=3.9.322,!=3.9.323,!=3.9.324,!=3.9.325,!=3.9.326,!=3.9.327,!=3.9.328,!=3.9.329,!=3.9.330,!=3.9.331,!=3.9.332,!=3.9.333,!=3.9.334,!=3.9.335,!=3.9.336,!=3.9.337,!=3.9.338,!=3.9.339,!=3.9.340,!=3.9.341,!=3.9.342,!=3.9.343,!=3.9.344,!=3.9.345,!=3.9.346,!=3.9.347,!=3.9.348,!=3.9.349,!=3.9.350,!=3.9.351,!=3.9.352,!=3.9.353,!=3.9.354,!=3.9.355,!=3.9.356,!=3.9.357,!=3.9.358,!=3.9.359,!=3.9.360,!=3.9.361,!=3.9.362,!=3.9.363,!=3.9.364,!=3.9.365,!=3.9.366,!=3.9.367,!=3.9.368,!=3.9.369,!=3.9.370,!=3.9.371,!=3.9.372,!=3.9.373,!=3.9.374,!=3.9.375,!=3.9.376,!=3.9.377,!=3.9.378,!=3.9.379,!=3.9.380,!=3.9.381,!=3.9.382,!=3.9.383,!=3.9.384,!=3.9.385,!=3.9.386,!=3.9.387,!=3.9.388,!=3.9.389,!=3.9.390,!=3.9.391,!=3.9.392,!=3.9.393,!=3.9.394,!=3.9.395,!=3.9.396,!=3.9.397,!=3.9.398,!=3.9.399,!=3.9.400,!=3.9.401,!=3.9.402,!=3.9.403,!=3.9.404,!=3.9.405,!=3.9.406,!=3.9.407,!=3.9.408,!=3.9.409,!=3.9.410,!=3.9.411,!=3.9.412,!=3.9.413,!=3.9.414,!=3.9.415,!=3.9.416,!=3.9.417,!=3.9.418,!=3.9.419,!=3.9.420,!=3.9.421,!=3.9.422,!=3.9.423,!=3.9.424,!=3.9.425,!=3.9.426,!=3.9.427,!=3.9.428,!=3.9.429,!=3.9.430,!=3.9.431,!=3.9.432,!=3.9.433,!=3.9.434,!=3.9.435,!=3.9.436,!=3.9.437,!=3.9.438,!=3.9.439,!=3.9.440,!=3.9.441,!=3.9.442,!=3.9.443,!=3.9.444,!=3.9.445,!=3.9.446,!=3.9.447,!=3.9.448,!=3.9.449,!=3.9.450,!=3.9.451,!=3.9.452,!=3.9.453,!=3.9.454,!=3.9.455,!=3.9.456,!=3.9.457,!=3.9.458,!=3.9.459,!=3.9.460,!=3.9.461,!=3.9.462,!=3.9.463,!=3.9.464,!=3.9.465,!=3.9.466,!=3.9.467,!=3.9.468,!=3.9.469,!=3.9.470,!=3.9.471,!=3.9.472,!=3.9.473,!=3.9.474,!=3.9.475,!=3.9.476,!=3.9.477,!=3.9.478,!=3.9.479,!=3.9.480,!=3.9.481,!=3.9.482,!=3.9.483,!=3.9.484,!=3.9.485,!=3.9.486,!=3.9.487,!=3.9.488,!=3.9.489,!=3.9.490,!=3.9.491,!=3.9.492,!=3.9.493,!=3.9.494,!=3.9.495,!=3.9.496,!=3.9.497,!=3.9.498,!=3.9.499,!=3.9.500,!=3.9.501,!=3.9.502,!=3.9.503,!=3.9.504,!=3.9.505,!=3.9.506,!=3.9.507,!=3.9.508,!=3.9.509,!=3.9.510,!=3.9.511,!=3.9.512,!=3.9.513,!=3.9.514,!=3.9.515,!=3.9.516,!=3.9.517,!=3.9.518,!=3.9.519,!=3.9.520,!=3.9.521,!=3.9.522,!=3.9.523,!=3.9.524,!=3.9.525,!=3.9.526,!=3.9.527,!=3.9.528,!=3.9.529,!=3.9.530,!=3.9.531,!=3.9.532,!=3.9.533,!=3.9.534,!=3.9.535,!=3.9.536,!=3.9.537,!=3.9.538,!=3.9.539,!=3.9.540,!=3.9.541,!=3.9.542,!=3.9.543,!=3.9.544,!=3.9.545,!=3.9.546,!=3.9.547,!=3.9.548,!=3.9.549,!=3.9.550,!=3.9.551,!=3.9.552,!=3.9.553,!=3.9.554,!=3.9.555,!=3.9.556,!=3.9.557,!=3.9.558,!=3.9.559,!=3.9.560,!=3.9.561,!=3.9.562,!=3.9.563,!=3.9.564,!=3.9.565,!=3.9.566,!=3.9.567,!=3.9.568,!=3.9.569,!=3.9.570,!=3.9.571,!=3.9.572,!=3.9.573,!=3.9.574,!=3.9.575,!=3.9.576,!=3.9.577,!=3.9.578,!=3.9.579,!=3.9.580,!=3.9.581,!=3.9.582,!=3.9.583,!=3.9.584,!=3.9.585,!=3.9.586,!=3.9.587,!=3.9.588,!=3.9.589,!=3.9.590,!=3.9.591,!=3.9.592,!=3.9.593,!=3.9.594,!=3.9.595,!=3.9.596,!=3.9.597,!=3.9.598,!=3.9.599,!=3.9.600,!=3.9.601,!=3.9.602,!=3.9.603,!=3.9.604,!=3.9.605,!=3.9.606,!=3.9.607,!=3.9.608,!=3.9.609,!=3.9.610,!=3.9.611,!=3.9.612,!=3.9.613,!=3.9.614,!=3.9.615,!=3.9.616,!=3.9.617,!=3.9.618,!=3.9.619,!=3.9.620,!=3.9.621,!=3.9.622,!=3.9.623,!=3.9.624,!=3.9.625,!=3.9.626,!=3.9.627,!=3.9.628,!=3.9.629,!=3.9.630,!=3.9.631,!=3.9.632,!=3.9.633,!=3.9.634,!=3.9.635,!=3.9.636,!=3.9.637,!=3.9.638,!=3.9.639,!=3.9.640,!=3.9.641,!=3.9.642,!=3.9.643,!=3.9.644,!=3.9.645,!=3.9.646,!=3.9.647,!=3.9.648,!=3.9.649,!=3.9.650,!=3.9.651,!=3.9.652,!=3.9.653,!=3.9.654,!=3.9.655,!=3.9.656,!=3.9.657,!=3.9.658,!=3.9.659,!=3.9.660,!=3.9.661,!=3.9.662,!=3.9.663,!=3.9.664,!=3.9.665,!=3.9.666,!=3.9.667,!=3.9.668,!=3.9.669,!=3.9.670,!=3.9.671,!=3.9.672,!=3.9.673,!=3.9.674,!=3.9.675,!=3.9.676,!=3.9.677,!=3.9.678,!=3.9.679,!=3.9.680,!=3.9.681,!=3.9.682,!=3.9.683,!=3.9.684,!=3.9.685,!=3.9.686,!=3.9.687,!=3.9.688,!=3.9.689,!=3.9.690,!=3.9.691,!=3.9.692,!=3.9.693,!=3.9.694,!=3.9.695,!=3.9.696,!=3.9.697,!=3.9.698,!=3.9.699,!=3.9.700,!=3.9.701,!=3.9.702,!=3.9.703,!=3.9.704,!=3.9.705,!=3.9.706,!=3.9.707,!=3.9.708,!=3.9.709,!=3.9.710,!=3.9.711,!=3.9.712,!=3.9.713,!=3.9.714,!=3.9.715,!=3.9.716,!=3.9.717,!=3.9.718,!=3.9.719,!=3.9.720,!=3.9.721,!=3.9.722,!=3.9.723,!=3.9.724,!=3.9.725,!=3.9.726,!=3.9.727,!=3.9.728,!=3.9.729,!=3.9.730,!=3.9.731,!=3.9.732,!=3.9.733,!=3.9.734,!=3.9.735,!=3.9.736,!=3.9.737,!=3.9.738,!=3.9.739,!=3.9.740,!=3.9.741,!=3.9.742,!=3.9.743,!=3.9.744,!=3.9.745,!=3.9.746,!=3.9.747,!=3.9.748,!=3.9.749,!=3.9.750,!=3.9.751,!=3.9.752,!=3.9.753,!=3.9.754,!=3.9.755,!=3.9.756,!=3.9.757,!=3.9.758,!=3.9.759,!=3.9.760,!=3.9.761,!=3.9.762,!=3.9.763,!=3.9.764,!=3.9.765,!=3.9.766,!=3.9.767,!=3.9.768,!=3.9.769,!=3.9.770,!=3.9.771,!=3.9.772,!=3.9.773,!=3.9.774,!=3.9.775,!=3.9.776,!=3.9.777,!=3.9.778,!=3.9.779,!=3.9.780,!=3.9.781,!=3.9.782,!=3.9.783,!=3.9.784,!=3.9.785,!=3.9.786,!=3.9.787,!=3.9.788,!=3.9.789,!=3.9.790,!=3.9.791,!=3.9.792,!=3.9.793,!=3.9.794,!=3.9.795,!=3.9.796,!=3.9.797,!=3.9.798,!=3.9.799,!=3.9.800,!=3.9.801,!=3.9.802,!=3.9.803,!=3.9.804,!=3.9.805,!=3.9.806,!=3.9.807,!=3.9.808,!=3.9.809,!=3.9.810,!=3.9.811,!=3.9.812,!=3.9.813,!=3.9.814,!=3.9.815,!=3.9.816,!=3.9.817,!=3.9.818,!=3.9.819,!=3.9.820,!=3.9.821,!=3.9.822,!=3.9.823,!=3.9.824,!=3.9.825,!=3.9.826,!=3.9.827,!=3.9.828,!=3.9.829,!=3.9.830,!=3.9.831,!=3.9.832,!=3.9.833,!=3.9.834,!=3.9.835,!=3.9.836,!=3.9.837,!=3.9.838,!=3.9.839,!=3.9.840,!=3.9.841,!=3.9.842,!=3.9.843,!=3.9.844,!=3.9.845,!=3.9.846,!=3.9.847,!=3.9.848,!=3.9.849,!=3.9.850,!=3.9.851,!=3.9.852,!=3.9.853,!=3.9.854,!=3.9.855,!=3.9.856,!=3.9.857,!=3.9.858,!=3.9.859,!=3.9.860,!=3.9.861,!=3.9.862,!=3.9.863,!=3.9.864,!=3.9.865,!=3.9.866,!=3.9.867,!=3.9.868,!=3.9.869,!=3.9.870,!=3.9.871,!=3.9.872,!=3.9.873,!=3.9.874,!=3.9.875,!=3.9.876,!=3.9.877,!=3.9.878,!=3.9.879,!=3.9.880,!=3.9.881,!=3.9.882,!=3.9.883,!=3.9.884,!=3.9.885,!=3.9.886,!=3.9.887,!=3.9.888,!=3.9.889,!=3.9.890,!=3.9.891,!=3.9.892,!=3.9.893,!=3.9.894,!=3.9.895,!=3.9.896,!=3.9.897,!=3.9.898,!=3.9.899,!=3.9.900,!=3.9.901,!=3.9.902,!=3.9.903,!=3.9.904,!=3.9.905,!=3.9.906,!=3.9.907,!=3.9.908,!=3.9.909,!=3.9.910,!=3.9.911,!=3.9.912,!=3.9.913,!=3.9.914,!=3.9.915,!=3.9.916,!=3.9.917,!=3.9.918,!=3.9.919,!=3.9.920,!=3.9.921,!=3.9.922,!=3.9.923,!=3.9.924,!=3.9.925,!=3.9.926,!=3.9.927,!=3.9.928,!=3.9.929,!=3.9.930,!=3.9.931,!=3.9.932,!=3.9.933,!=3.9.934,!=3.9.935,!=3.9.936,!=3.9.937,!=3.9.938,!=3.9.939,!=3.9.940,!=3.9.941,!=3.9.942,!=3.9.943,!=3.9.944,!=3.9.945,!=3.9.946,!=3.9.947,!=3.9.948,!=3.9.949,!=3.9.950,!=3.9.951,!=3.9.952,!=3.9.953,!=3.9.954,!=3.9.955,!=3.9.956,!=3.9.957,!=3.9.958,!=3.9.959,!=3.9.960,!=3.9.961,!=3.9.962,!=3.9.963,!=3.9.964,!=3.9.965,!=3.9.966,!=3.9.967,!=3.9.968,!=3.9.969,!=3.9.970,!=3.9.971,!=3.9.972,!=3.9.973,!=3.9.974,!=3.9.975,!=3.9.976,!=3.9.977,!=3.9.978,!=3.9.979,!=3.9.980,!=3.9.981,!=3.9.982,!=3.9.983,!=3.9.984,!=3.9.985,!=3.9.986,!=3.9.987,!=3.9.988,!=3.9.989,!=3.9.990,!=3.9.991,!=3.9.992,!=3.9.993,!=3.9.994,!=3.9.995,!=3.9.996,!=3.9.997,!=3.9.998,!=3.9.999,!=3.9.1000,!=3.9.1001,!=3.9.1002,!=3.9.1003,!=3.9.1004,!=3.9.1005,!=3.9.1006,!=3.9.1007,!=3.9.1008,!=3.9.1009,!=3.9.1010,!=3.9.1011,!=3.9.1012,!=3.9.1013,!=3.9.1014,!=3.9.1015,!=3.9.1016,!=3.9.1017,!=3.9.1018,!=3.9.1019,!=3.9.1020,!=3.9.1021,!=3.9.1022,!=3.9.1023,!=3.9.1024,!=3.9.1025,!=3.9.1026,!=3.9.1027,!=3.9.1028,!=3.9.1029,!=3.9.1030,!=3.9.1031,!=3.9.1032,!=3.9.1033,!=3.9.1034,!=3.9.1035,!=3.9.1036,!=3.9.1037,!=3.9.1038,!=3.9.1039,!=3.9.1040,!=3.9.1041,!=3.9.1042,!=3.9.1043,!=3.9.1044,!=3.9.1045,!=3.9.1046,!=3.9.1047,!=3.9.1048,!=3.9.1049,!=3.9.1050,!=3.9.1051,!=3.9.1052,!=3.9.1053,!=3.9.1054,!=3.9.1055,!=3.9.1056,!=3.9.1057,!=3.9.1058,!=3.9.1059,!=3.9.1060,!=3.9.1061,!=3.9.1062,!=3.9.1063,!=3.9.1064,!=3.9.1065,!=3.9.1066,!=3.9.1067,!=3.9.1068,!=3.9.1069,!=3.9.1070,!=3.9.1071,!=3.9.1072,!=3.9.1073,!=3.9.1074,!=3.9.1075,!=3.9.1076,!=3.9.1077,!=3.9.1078,!=3.9.1079,!=3.9.1080,!=3.9.1081,!=3.9.1082,!=3.9.1083,!=3.9.1084,!=3.9.1085,!=3.9.1086,!=3.9.1087,!=3.9.1088,!=3.9.1089,!=3.9.1090,!=3.9.1091,!=3.9.1092,!=3.9.1093,!=3.9.1094,!=3.9.1095,!=3.9.1096,!=3.9.1097,!=3.9.1098,!=3.9.1099,!=3.9.1100,!=3.9.1101,!=3.9.1102,!=3.9.1103,!=3.9.1104,!=3.9.1105,!=3.9.1106,!=3.9.1107,!=3.9.1108,!=3.9.1109,!=3.9.1110,!=3.9.1111,!=3.9.1112,!=3.9.1113,!=3.9.1114,!=3.9.1115,!=3.9.1116,!=3.9.1117,!=3.9.1118,!=3.9.1119,!=3.9.1120,!=3.9.1121,!=3.9.1122,!=3.9.1123,!=3.9.1124,!=3.9.1125,!=3.9.1126,!=3.9.1127,!=3.9.1128,!=3.9.1129,!=3.9.1130,!=3.9.1131,!=3.9.1132,!=3.9.1133,!=3.9.1134,!=3.9.1135,!=3.9.1136,!=3.9.1137,!=3.9.1138,!=3.9.1139,!=3.9.1140,!=3.9.1141,!=3.9.1142,!=3.9.1143,!=3.9.1144,!=3.9.1145,!=3.9.1146,!=3.9.1147,!=3.9.1148,!=3.9.1149,!=3.9.1150,!=3.9.1151,!=3.9.1152,!=3.9.1153,!=3.9.1154,!=3.9.1155,!=3.9.1156,!=3.9.1157,!=3.9.1158,!=3.9.1159,!=3.9.1160,!=3.9.1161,!=3.9.1162,!=3.9.1163,!=3.9.1164,!=3.9.1165,!=3.9.1166,!=3.9.1167,!=3.9.1168,!=3.9.1169,!=3.9.1170,!=3.9.1171,!=3.9.1172,!=3.9.1173,!=3.9.1174,!=3.9.1175,!=3.9.1176,!=3.9.1177,!=3.9.1178,!=3.9.1179,!=3.9.1180,!=3.9.1181,!=3.9.1182,!=3.9.1183,!=3.9.1184,!=3.9.1185,!=3.9.1186,!=3.9.1187,!=3.9.1188,!=3.9.1189,!=3.9.1190,!=3.9.1191,!=3.9.1192,!=3.9.1193,!=3.9.1194,!=3.9.1195,!=3.9.1196,!=3.9.1197,!=3.9.1198,!=3.9.1199,!=3.9.1200,!=3.9.1201,!=3.9.1202,!=3.9.1203,!=3.9.1204,!=3.9.1205,!=3.9.1206,!=3.9.1207,!=3.9.1208,!=3.9.1209,!=3.9.1210,!=3.9.1211,!=3.9.1212,!=3.9.1213,!=3.9.1214,!=3.9.1215,!=3.9.1216,!=3.9.1217,!=3.9.1218,!=3.9.1219,!=3.9.1220,!=3.9.1221,!=3.9.1222,!=3.9.1223,!=3.9.1224,!=3.9.1225,!=3.9.1226,!=3.9.1227,!=3.9.1228,!=3.9.1229,!=3.9.1230,!=3.9.1231,!=3.9.1232,!=3.9.1233,!=3.9.1234,!=3.9.1235,!=3.9.1236,!=3.9.1237,!=3.9.1238,!=3.9.1239,!=3.9.1240,!=3.9.1241,!=3.9.1242,!=3.9.1243,!=3.9.1244,!=3.9.1245,!=3.9.1246,!=3.9.1247,!=3.9.1248,!=3.9.1249,!=3.9.1250,!=3.9.1251,!=3.9.1252,!=3.9.1253,!=3.9.1254,!=3.9.1255,!=3.9.1256,!=3.9.1257,!=3.9.1258,!=3.9.1259,!=3.9.1260,!=3.9.1261,!=3.9.1262,!=3.9.1263,!=3.9.1264,!=3.9.1265,!=3.9.1266,!=3.9.1267,!=3.9.1268,!=3.9.1269,!=3.9.1270,!=3.9.1271,!=3.9.1272,!=3.9.1273,!=3.9.1274,!=3.9.1275,!=3.9.1276,!=3.9.1277,!=3.9.1278,!=3.9.1279,!=3.9.1280,!=3.9.1281,!=3.9.1282,!=3.9.1283,!=3.9.1284,!=3.9.1285,!=3.9.1286,!=3.9.1287,!=3.9.1288,!=3.9.1289,!=3.9.1290,!=3.9.1291,!=3.9.1292,!=3.9.1293,!=3.9.1294,!=3.9.1295,!=3.9.1296,!=3.9.1297,!=3.9.1298,!=3.9.1299,!=3.9.1300,!=3.9.1301,!=3.9.1302,!=3.9.1303,!=3.9.1304,!=3.9.1305,!=3.9.1306,!=3.9.1307,!=3.9.1308,!=3.9.1309,!=3.9.1310,!=3.9.1311,!=3.9.1312,!=3.9.1313,!=3.9.1314,!=3.9.1315,!=3.9.1316,!=3.9.1317,!=3.9.1318,!=3.9.1319,!=3.9.1320,!=3.9.1321,!=3.9.1322,!=3.9.1323,!=3.9.1324,!=3.9.1325,!=3.9.1326,!=3.9.1327,!=3.9.1328,!=3.9.1329,!=3.9.1330,!=3.9.1331,!=3.9.1332,!=3.9.1333,!=3.9.1334,!=3.9.1335,!=3.9.1336,!=3.9.1337,!=3.9.1338,!=3.9.1339,!=3.9.1340,!=3.9.1341,!=3.9.1342,!=3.9.1343,!=3.9.1344,!=3.9.1345,!=3.9.1346,!=3.9.1347,!=3.9.1348,!=3.9.1349,!=3.9.1350,!=3.9.1351,!=3.9.1352,!=3.9.1353,!=3.9.1354,!=3.9.1355,!=3.9.1356,!=3.9.1357,!=3.9.1358,!=3.9.1359,!=3.9.1360,!=3.9.1361,!=3.9.1362,!=3.9.1363,!=3.9.1364,!=3.9.1365,!=3.9.1366,!=3.9.1367,!=3.9.1368,!=3.9.1369,!=3.9.1370,!=3.9.1371,!=3.9.1372,!=3.9.1373,!=3.9.1374,!=3.9.1375,!=3.9.1376,!=3.9.1377,!=3.9.1378,!=3.9.1379,!=3.9.1380,!=3.9.1381,!=3.9.1382,!=3.9.1383,!=3.9.1384,!=3.9.1385,!=3.9.1386,!=3.9.1387,!=3.9.1388,!=3.9.1389,!=3.9.1390,!=3.9.1391,!=3.9.1392,!=3.9.1393,!=3.9.1394,!=3.9.1395,!=3.9.1396,!=3.9.1397,!=3.9.1398,!=3.9.1399,!=3.9.1400,!=3.9.1401,!=3.9.1402,!=3.9.1403,!=3.9.1404,!=3.9.1405,!=3.9.1406,!=3.9.1407,!=3.9.1408,!=3.9.1409,!=3.9.1410,!=3.9.1411,!=3.9.1412,!=3.9.1413,!=3.9.1414,!=3.9.1415,!=3.9.1416,!=3.9.1417,!=3.9.1418,!=3.9.1419,!=3.9.1420,!=3.9.1421,!=3.9.1422,!=3.9.1423,!=3.9.1424,!=3.9.1425,!=3.9.1426,!=3.9.1427,!=3.9.1428,!=3.9.1429,!=3.9.1430,!=3.9.1431,!=3.9.1432,!=3.9.1433,!=3.9.1434,!=3.9.1435,!=3.9.1436,!=3.9.1437,!=3.9.1438,!=3.9.1439,!=3.9.1440,!=3.9.1441,!=3.9.1442,!=3.9.1443,!=3.9.1444,!=3.9.1445,!=3.9.1446,!=3.9.1447,!=3.9.1448,!=3.9.1449,!=3.9.1450,!=3.9.1451,!=3.9.1452,!=3.9.1453,!=3.9.1454,!=3.9.1455,!=3.9.1456,!=3.9.1457,!=3.9.1458,!=3.9.1459,!=3.9.1460,!=3.9.1461,!=3.9.1462,!=3.9.1463,!=3.9.1464,!=3.9.1465,!=3.9.1466,!=3.9.1467,!=3.9.1468,!=3.9.1469,!=3.9.1470,!=3.9.1471,!=3.9.1472,!=3.9.1473,!=3.9.1474,!=3.9.1475,!=3.9.1476,!=3.9.1477,!=3.9.1478,!=3.9.1479,!=3.9.1480,!=3.9.1481,!=3.9.1482,!=3.9.1483,!=3.9.1484,!=3.9.1485,!=3.9.1486,!=3.9.1487,!=3.9.1488,!=3.9.1489,!=3.9.1490,!=3.9.1491,!=3.9.1492,!=3.9.1493,!=3.9.1494,!=3.9.1495,!=3.9.1496,!=3.9.1497,!=3.9.1498,!=3.9.1499,!=3.9.1500,!=3.9.1501,!=3.9.1502,!=3.9.1503,!=3.9.1504,!=3.9.1505,!=3.9.1506,!=3.9.1507,!=3.9.1508,!=3.9.1509,!=3.9.1510,!=3.9.1511,!=3.9.1512,!=3.9.1513,!=3.9.1514,!=3.9.1515,!=3.9.1516,!=3.9.1517,!=3.9.1518,!=3.9.1519,!=3.9.1520,!=3.9.1521,!=3.9.1522,!=3.9.1523,!=3.9.1524,!=3.9.1525,!=3.9.1526,!=3.9.1527,!=3.9.1528,!=3.9.1529,!=3.9.1530,!=3.9.1531,!=3.9.1532,!=3.9.1533,!=3.9.1534,!=3.9.1535,!=3.9.1536,!=3.9.1537,!=3.9.1538,!=3.9.1539,!=3.9.1540,!=3.9.1541,!=3.9.1542,!=3.9.1543,!=3.9.1544,!=3.9.1545,!=3.9.1546,!=3.9.1547,!=3.9.1548,!=3.9.1549,!=3.9.1550,!=3.9.1551,!=3.9.1552,!=3.9.1553,!=3.9.1554,!=3.9.1555,!=3.9.1556,!=3.9.1557,!=3.9.1558,!=3.9.1559,!=3.9.1560,!=3.9.1561,!=3.9.1562,!=3.9.1563,!=3.9.1564,!=3.9.1565,!=3.9.1566,!=3.9.1567,!=3.9.1568,!=3.9.1569,!=3.9.1570,!=3.9.1571,!=3.9.1572,!=3.9.1573,!=3.9.1574,!=3.9.1575,!=3.9.1576,!=3.9.1577,!=3.9.1578,!=3.9.1579,!=3.9.1580
```