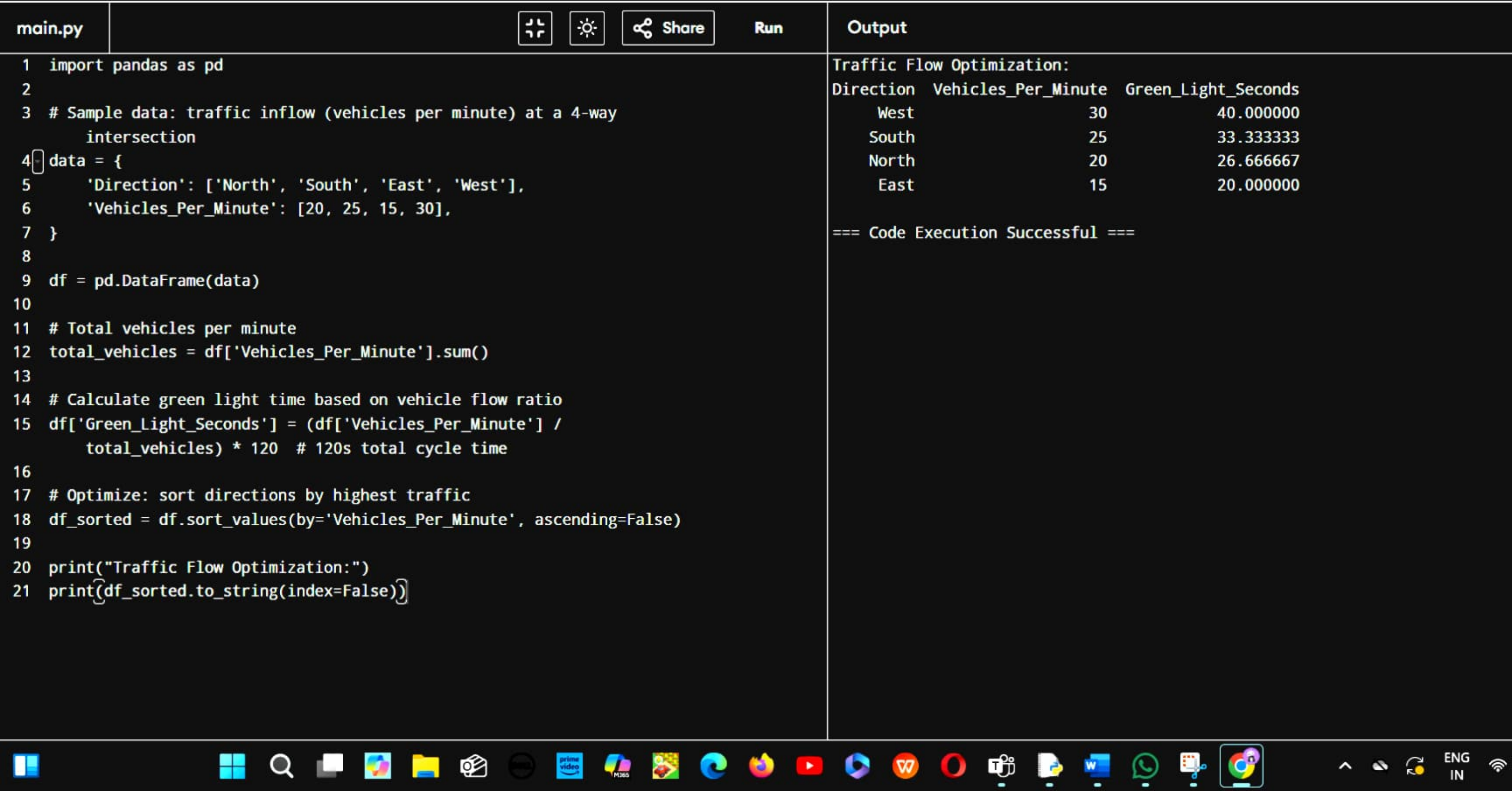
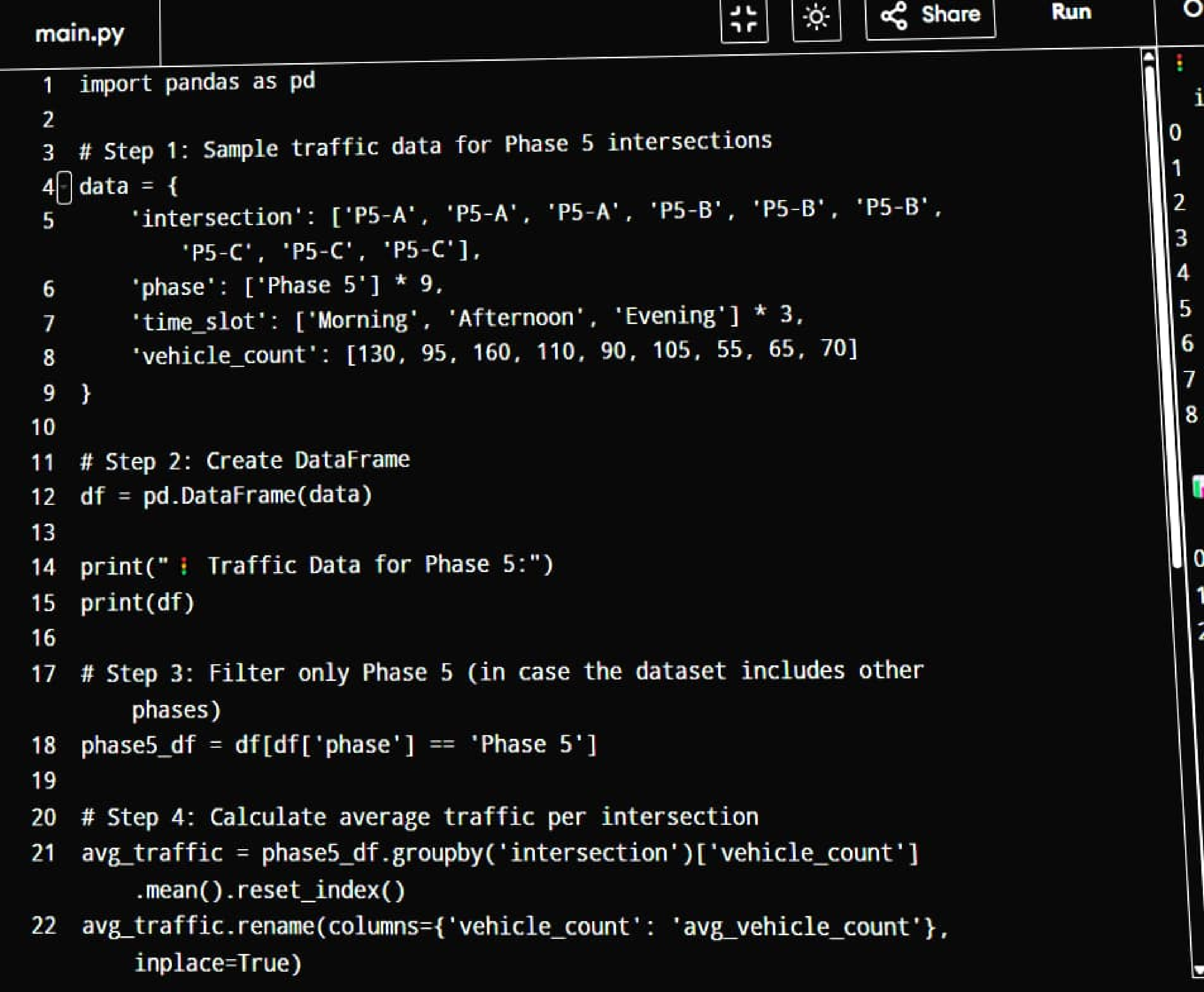
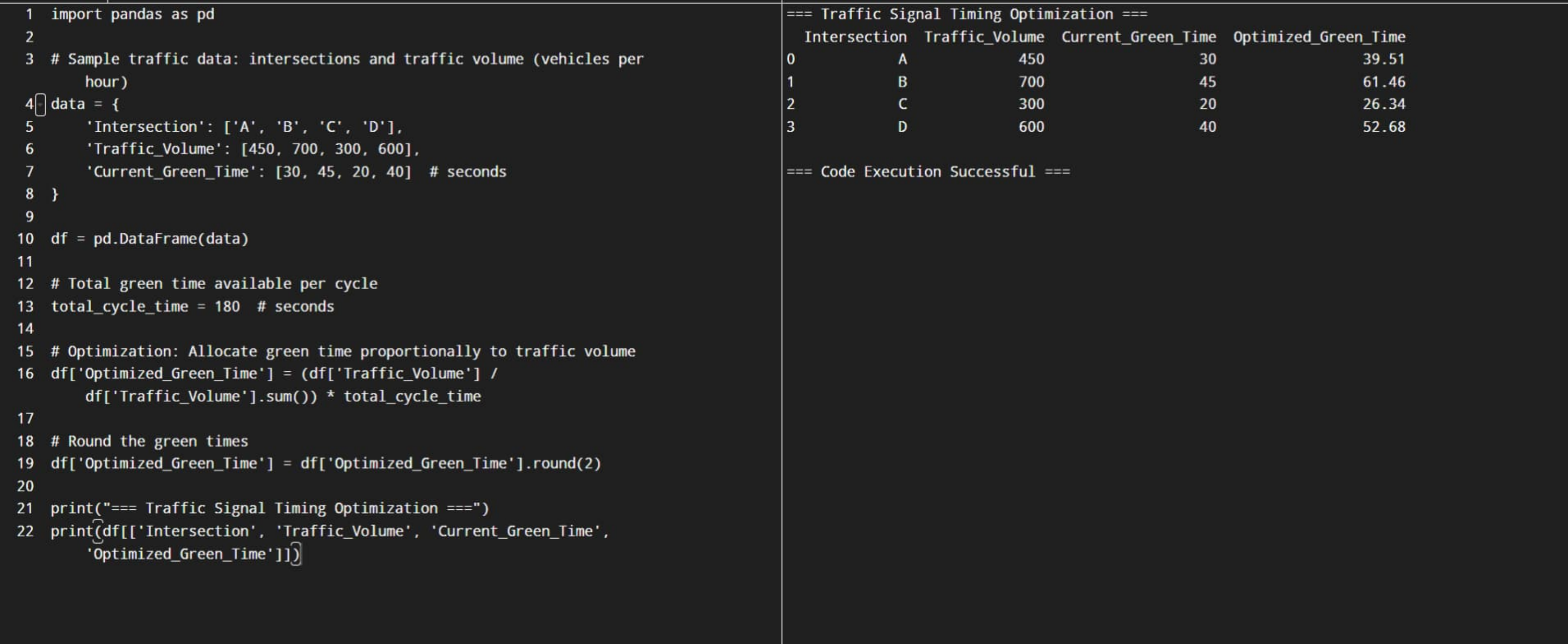


```python
import pandas as pd

# Sample data: traffic inflow (vehicles per minute) at a 4-way intersection
data = {
    'Direction': ['North', 'South', 'East', 'West'],
    'Vehicles_Per_Minute': [20, 25, 15, 30],
}

df = pd.DataFrame(data)

# Total vehicles per minute
total_vehicles = df['Vehicles_Per_Minute'].sum()

# Calculate green light time based on vehicle flow ratio
df['Green_Light_Seconds'] = (df['Vehicles_Per_Minute'] / total_vehicles) * 120  # 120s total cycle time

# Optimize: sort directions by highest traffic
df_sorted = df.sort_values(by='Vehicles_Per_Minute', ascending=False)

print("Traffic Flow Optimization:")
print(df_sorted.to_string(index=False))
```

```
Traffic Flow Optimization:
Direction  Vehicles_Per_Minute  Green_Light_Seconds
     West                   30            40.000000
    South                   25            33.333333
    North                   20            26.666667
     East                   15            20.000000

=== Code Execution Successful ===
```



```python
import pandas as pd

# Step 1: Sample traffic data for Phase 5 intersections
data = {
    'intersection': ['P5-A', 'P5-A', 'P5-A', 'P5-B', 'P5-B', 'P5-B', 'P5-C', 'P5-C', 'P5-C'],
    'phase': ['Phase 5'] * 9,
    'time_slot': ['Morning', 'Afternoon', 'Evening'] * 3,
    'vehicle_count': [130, 95, 160, 110, 90, 105, 55, 65, 70]
}

# Step 2: Create DataFrame
df = pd.DataFrame(data)

print("📊 Traffic Data for Phase 5:")
print(df)

# Step 3: Filter only Phase 5 (in case the dataset includes other phases)
phase5_df = df[df['phase'] == 'Phase 5']

# Step 4: Calculate average traffic per intersection
avg_traffic = phase5_df.groupby('intersection')['vehicle_count'].mean().reset_index()
avg_traffic.rename(columns={'vehicle_count': 'avg_vehicle_count'}, inplace=True)
```



```python
import pandas as pd

# Sample traffic data: intersections and traffic volume (vehicles per hour)
data = {
    'Intersection': ['A', 'B', 'C', 'D'],
    'Traffic_Volume': [450, 700, 300, 600],
    'Current_Green_Time': [30, 45, 20, 40]  # seconds
}

df = pd.DataFrame(data)

# Total green time available per cycle
total_cycle_time = 180  # seconds

# Optimization: Allocate green time proportionally to traffic volume
df['Optimized_Green_Time'] = (df['Traffic_Volume'] / df['Traffic_Volume'].sum()) * total_cycle_time

# Round the green times
df['Optimized_Green_Time'] = df['Optimized_Green_Time'].round(2)

print("=== Traffic Signal Timing Optimization ===")
print(df[['Intersection', 'Traffic_Volume', 'Current_Green_Time', 'Optimized_Green_Time']])
```

```
=== Traffic Signal Timing Optimization ===
  Intersection  Traffic_Volume  Current_Green_Time  Optimized_Green_Time
0            A             450                  30                 39.51
1            B             700                  45                 61.46
2            C             300                  20                 26.34
3            D             600                  40                 52.68

=== Code Execution Successful ===
```