Pseudo-code

```
1.  Display Welcome Message
2.  Print "Welcome to the Traffic Analysis Program!"
3.  Loop Until User Decides to Exit
4.  Prompt the user to enter the date of the traffic survey (validate_date_input()):
5.  Validate Day: Ensure the day is between 1 and 31.
6.  Validate Month: Ensure the month is between 1 and 12.
7.  Validate Year: Ensure the year is between 2000 and 2024.
8.  Use the entered date to define the file path for traffic data (define_file_path()).
9.  Process Traffic Data
10. Call process_csv_data() with the file path and various traffic-related counters:
11. Read CSV File: Open and read the file using csv.DictReader.
12. Iterate Through Rows: Process each row:
13. Increment total_vehicles for each row.
14. Check and categorize vehicle types:
15. Truck: Increment total_trucks.
16. Scooter, Motorcycle, Bicycle: Update respective counters.
17. Count vehicles turning and speeding based on direction_in and speed attributes.
18. Count vehicles passing through specific junctions (Elm Avenue/Rabbit Road or Hanley
    Highway/Westway).
19. Track hourly data (per_hour_vehicles) and rainy hours.
20. Calculate additional statistics:
21. Percentage of trucks.
22. Average bikes per hour.
23. Scooter percentage in Elm Avenue/Rabbit Road.
24. Peak time using find_peak_time().
25. Display results using display_outcomes():
26. Print formatted traffic analysis results for the selected date.
27. Save the results to a file (save_results_to_file()).
28. Display Histogram
29. Read the CSV file again to extract traffic data.
30. Initialize the HistogramApp with the traffic data and date:
31. Process Hourly Data: Separate counts for Elm Avenue/Rabbit Road and Hanley
    Highway/Westway.
32. Setup Window: Create a graphical window.
33. Draw Title and Legend: Add the histogram title and legend for junctions.
34. Draw Bars: Plot hourly traffic counts as bars for both junctions.
35. Draw Axes and Labels: Label the x-axis with hours and y-axis values.
36. Wait for user input before closing the histogram window.
```

37. Prompt User to Continue or Exit
38. Ask if the user wants to analyze data for another date (validate_continue_input()):
39. If "Yes," repeat the process.
40. If "No," exit the program with a goodbye message.
41. Functions
42. validate_date_input()
43. Validate and return user input for day, month, and year.
44. define_file_path(date)
45. Generate the file path for traffic data based on the input date.
46. process_csv_data()
47. Process traffic data from the CSV file:
48. Count total vehicles, trucks, bicycles, scooters, and others.
49. Calculate various statistics (e.g., peak time, percentage of scooters).
50. Track vehicles through junctions and their behaviors.
51. display_outcomes()
52. Display processed statistics in a readable format.
53. save_results_to_file()
54. Append traffic analysis results to a text file.
55. HistogramApp
56. Attributes:
57. Traffic data and selected date.
58. Methods:
59. process_hourly_data(): Prepare data for each hour and junction.
60. setup_window(): Configure the graphical window.
61. draw_title_and_legend(): Add title and legend for the histogram.
62. draw_bars(): Draw bars representing traffic counts for each hour.
63. draw_axes_and_labels(): Add axes and labels for clarity.
64. display(): Render and display the histogram.
65. validate_continue_input()
66. Prompt the user to decide whether to analyze another date's data.
67. find_peak_time()
68. Determine the hour(s) with the maximum traffic and format the result.
69. Flow Diagram
70. User Input: Date → File Path
71. File Processing:
72. Read Data → Analyze → Display Results → Save Results.
73. Histogram Visualization:
74. Extract Data → Render Histogram.
75. User Decision: Continue or Exit