Design patterns for Question 1:

To solve this question we can use strategy pattern.To determine leader and follower we need to use haversine method.After determining distance between the two pairs of trajectories at each time step the vehicle which is leading and which is following can be calculated.The vehicle with a smaller distance to the other vehicle is considered the leader.

Steps:

- ☐ Develop a function to calculate haversine distance, which takes Latitude and Longitude of two pairs and return calculated distance.
- ☐ Read all files using python.
- ☐ Declare a list for all pair of trajectories.
- ☐ Determine which distance is smaller than other and by using that, declare leader and follower.

Test Plan:

1.Test the function haversine(lat1, lon1, lat2, lon2) to check if it calculates the distance between two given points correctly.

Test cases:

- ●Test with the same point (lat1, lon1) and (lat2, lon2) and check if the distance is zero.
- ●Test with two points on the same latitude and different longitudes and verify the distance calculated by the function.
- ●Test with two points on the same longitude and different latitudes and verify the distance calculated by the function.
- ●Test with two points having different latitudes and longitudes and verify the distance calculated by the function.

2.Test the reading of the trajectory files.

Test cases:

- ●Test with an existing file.
- ●Test with a non-existing file and verify if it raises the FileNotFoundError.

3.Test the pairing of the trajectories.

Test cases:

- ●Test with a pair of trajectories (T1, T2) and check if they are paired correctly.
- ●Test with a pair of trajectories (T1, T2_2) and check if they are paired correctly.
- ●Test with a pair of trajectories (T3, T4) and check if they are paired correctly.
- ●Test with a non-existing pair of trajectories and verify if it raises the IndexError.

4.Test the calculation of the leader and follower for each pair of trajectories.

Test cases:

- ●Test with a pair of trajectories (T1, T2) where T1 is the leader and T2 is the follower.
- ●Test with a pair of trajectories (T1, T2) where T2 is the leader and T1 is the follower.
- ●Test with a pair of trajectories (T1, T2) where both trajectories have the same coordinates, and check if it raises the ZeroDivisionError.
- ●Test with a pair of trajectories (T1, T2) where no matching time step exists, and verify if it prints the "No matching time step found for pair" message.
- ●Test with a pair of trajectories (T1, T2) where the distance is less than 3 meters (vehicle length) and verify if it selects the leader and follower correctly.
- ●Test with a pair of trajectories (T1, T2) where the distance is greater than 3 meters (vehicle length) and verify if it selects the leader and follower correctly.
- ●Test with a pair of trajectories (T1, T2) where there are multiple matching time steps and verify if it selects the leader and follower correctly.
- ●Test with a pair of trajectories (T1, T2) where one trajectory has more data points than the other and verify if it selects the leader and follower correctly.

5.Test the output of the leader and follower for each pair of trajectories.

Test cases:

- ●Test with a pair of trajectories (T1, T2) and check if the leader and follower are printed correctly.
- ●Test with a pair of trajectories (T1, T2_2) and check if the leader and follower are printed correctly.

●Test with a pair of trajectories (T3, T4) and check if the leader and follower are printed correctly.

Solution for Question 2:

Design Pattern:
To determine the minimum TTC between two pairs of trajectories, we need to follow these steps:
1.Load the trajectory data from the provided csv files using a data analysis library like pandas in Python.
2.Convert the latitude and longitude values into x and y coordinates using a suitable projection (e.g. Mercator) to calculate the distance between two points on the earth's surface.
3.Calculate the positions and speeds of the front bumpers of the vehicles for each time step using the provided data and assuming a vehicle length of 3 meters.
4.Compute the TTC between each pair of trajectories at each time step using the provided formula.
5.Find the minimum TTC value for each pair of trajectories and the corresponding time step.

Test Plan for Trajectory TTC Calculation:
1.Test Case for Loading Data:
- ●Input: Trajectory files (T1.csv, T2.csv, T2_2.csv, T3.csv, T4.csv)
- ●Output: Loaded dataframes T1, T2, T2_2, T3, T4

Procedure:
- a. Load each trajectory CSV file using pd.read_csv()
- b. Check if the data is loaded successfully by checking the shape and values of each dataframe

2.Test Case for Calculating Front Bumper Positions:
- ●Input: Dataframes T1, T2, T2_2, T3, T4
- ●Output: Calculated positions of front bumpers in each trajectory

Procedure:
- a. Calculate the distance between each data point in a trajectory and the first data point using the formula provided in the code (using numpy)
- b. Add the calculated distance values as a new column 'X' in each dataframe

c. Check if the calculated values are correct by comparing them with the provided example dataset

3.Test Case for Calculating Vehicle Speeds:

●Input: Dataframes T1, T2, T2_2, T3, T4

●Output: Calculated speeds of vehicles in each trajectory

Procedure:

a. Calculate the speed of each vehicle in a trajectory by dividing the difference in 'X' values by the difference in 'Time (s)' values (using numpy)

b. Add the calculated speed values as a new column 'V' in each dataframe

c. Check if the calculated values are correct by comparing them with the provided example dataset

4.Test Case for Calculating TTC Between T1 and T2:

●Input: Dataframes T1, T2, vehicle_length

●Output: Calculated TTC between T1 and T2

Procedure:

a. Loop through the data points of T1 and T2 (up to the length of the shortest trajectory)

b. Check if the current data point of T1 is behind the current data point of T2

c. If T1 is behind T2, calculate the TTC using the provided formula and append the value to a list of TTCs

d. Take the minimum value from the list of TTCs as the final TTC

e. Check if the calculated TTC is correct by manually verifying the calculation for a few data points

5.Test Case for Calculating TTC Between T1 and T2_2:

●Input: Dataframes T1, T2_2, vehicle_length

●Output: Calculated TTC between T1 and T2_2

Procedure:

a. Loop through the data points of T1 and T2_2 (up to the length of the shortest trajectory)

b. Check if the current data point of T1 is behind the current data point of T2_2

c. If T1 is behind T2_2, calculate the TTC using the provided formula and append the value to a list of TTCs

d. Take the minimum value from the list of TTCs as the final TTC

e. Check if the calculated TTC is correct by manually verifying the calculation for a few data points

6.Test Case for Calculating TTC Between T3 and T4:

- Input: Dataframes T3, T4, vehicle_length
- Output: Calculated TTC between T3 and T4

Procedure:

a. Loop through the data points of T3 and T4 (up to the length of the shortest trajectory)

b. Check if the current speed of T4 is greater than the current speed of T3 and if T4