```java
public class Vehicle {
   private int destination;
   private int bornTime;
   private int exitTime;

   public Vehicle(int destination, int bornTime) {
      this.destination = destination;
      this.bornTime = bornTime;
   }

   public String toString() {
      return "(" + destination + "," + bornTime + "," + exitTime + ")";
   }


   public int getDestination() {
      return destination;
   }

   public int getBornTime() {
      return bornTime;
   }

   public void setExitTime(int exitTime) {
      if (exitTime<bornTime) {
         throw new RuntimeException("exitTime must be > borntime");
      }
      this.exitTime = exitTime;
   }

   public int getSpentTime() {
      return exitTime - bornTime;
   }

   public static void main(String[] args) {
      int destination = 2;
      int time = 8;
      Vehicle v = new Vehicle(destination, time);
      System.out.println("Created vehicle v    : " + v);
      System.out.println("Destination          : " + v.getDestination());
      System.out.println("Born time            : " + v.getBornTime());
      System.out.println("Spent time           : " + v.getSpentTime());
      v.setExitTime(15);
      System.out.println("v after setting exit time: " + v);
      System.out.println("Spent time           : " + v.getSpentTime());
      v.setExitTime(1);    // Illegal setting - should cause error message
   }
}
```

```java
import java.util.ArrayList;

public class VehicleCollection {

  private ArrayList<Vehicle>theCollection;

  /*** Uppgift ***/

  private float meanTime;
  private int maxTime;
  private int numberMaxTime;

  /*** Uppgift ***/
  public VehicleCollection() {
    theCollection = new ArrayList<Vehicle>();
  }

  /*** Uppgift ***/
  public void add(Vehicle v) {
    theCollection.add(v);
  }

  /*** Uppgift ***/
  public String toString() {
    return theCollection.toString();
  }

//  public void clear() {
//     theCollection.clear();
//  }
//
  /***
   * Uppgift:
   * Computes the statistics for the stored vehicles i e
   * copmpute the mean and max value of time spent
   * and the number of vehicles using the max time
   ***/
  public void computeStatistics() {
    int sum = 0;
    maxTime = 0;
    for (int i=0; i<theCollection.size(); i++) {
      Vehicle v = theCollection.get(i);
      int time = v.getSpentTime();
      sum += time;
      maxTime = Math.max(maxTime, time);
    }
    meanTime = (float)sum/theCollection.size();
    numberMaxTime = 0;
    for (int i=0; i<theCollection.size(); i++) {
       Vehicle v = theCollection.get(i);
       if (v.getSpentTime() == maxTime) {
          numberMaxTime++;
       }
    }
  }

  /*
   * Prints the statistics for the stored vehicles
   */
  public void print() {
     System.out.println("Number of vehicles      :" + theCollection.size());
     System.out.println("Mean time               :" + meanTime);
     System.out.println("Max time                :" + maxTime);
     System.out.println("Number vehicles with max time: " + numberMaxTime);
     System.out.println("The collection          :" + theCollection);
  }

  public static void main(String[] args) {
     VehicleCollection vc = new VehicleCollection();
```

```java
     Vehicle v = new Vehicle(1,1);
     v.setExitTime(3);
     vc.add(v);
     v = new Vehicle(0, 2);
     v.setExitTime(3);
     vc.add(v);
     v = new Vehicle(0, 4);
     v.setExitTime(6);
     vc.add(v);
     System.out.println("\nPrint before statistics is computed:");
     vc.print();
     vc.computeStatistics();
     System.out.println("\nPrint after statistics is computed:");
     vc.print();

  }

}
```

```java
import java.util.Scanner;
import java.util.ArrayList;

public class Simulation {

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Number of destinations: ");
      int numberOfDests = sc.nextInt();
      System.out.print("Number of time steps  : ");
      int numberOfTimeSteps =sc.nextInt();

      TrafficSystem ts = new TrafficSystem(numberOfDests);

      /*** Tentamesuppgift
       * Create an array with one vehicleCollection for each destination
       ***/
      VehicleCollection[] out = new VehicleCollection[numberOfDests];
      for (int i = 0; i<numberOfDests; i++)  {
         out[i] = new VehicleCollection();
      }

      /*** Tentamensuppgift:
       * Perform the specified number of time steps and
       * store the vehicles coming out in the deignated
       * vehicle collection (i.e. a vehicle with destination 0
       * should be stored  on position 0 in trhe array,
       * a vehicle with destination 1 on position 1 etc
       ***/
      for (int i= 0; i<numberOfTimeSteps; i++) {
         Vehicle v = ts.step();
         if (v!=null) {
            out[v.getDestination()].add(v);
         }
         /* end uppgift */
      }

      System.out.println("\nStatistics:");
      for (int i=0; i<numberOfDests; i++) {
         System.out.println("\nDestination ’" + (char) (i + ’a’) +"’");
         System.out.println("===============");

         /*** Tentamensuppgift ? ***/
         out[i].computeStatistics();
         out[i].print();
         System.out.println("Lane: \n" + out[i].toString());
         /*** hit ***/
      }

      System.out.println("\nLeft in the system:");
      System.out.println(ts.toString());
   }
}
```