**Algorithm for seatbelt class:-**

1. Begin.
2. Define initial value of engine as 1 and randomly generate value of seat as 1 or 0.
3. Then generate two random serial numbers and call reporting agent class by creating its object and send two randomly generated serial numbers to that class.
4. Call violated\_agent and reciever\_agent function and get random violated and reporting agent id respectively.
5. Give a buffer time of 10 min and then start a while loop which will run till the engine is not equal to 0.
6. Check if the randomly generated value of the seat is 1 or 0. If it's 1 then go to step 6 or else go to step 10.
7. If the value of the seat is 1 then it means that the driver is wearing the seatbelt.
8. Give buffer time of 15min and randomly generate value for seat and engine.
9. If engine is 0 then terminate the loop and go to step 18.
10. If engine is 1 then repeat step 5 to 8.
11. If the value of the seat is 0 then it means that the driver is not wearing the seatbelt.
12. Give an alert of 5 beeps as warning to wear the seat belt and after warning provide a buffer time of 10min.
13. Again generate random value for engine and seat. If the value of the engine is 0 then it means the engine is stopped and thus terminate the loop and go to step 18.
14. If engine ==1 then check if the seat is 1 or 0. If it is 1 then repeat step 6 to 9 else go to step 14.
15. If the value of the seat is still 0 even after giving warning then it means that seatbelt violation is committed by the driver. Call send() function of reporting\_agent class by passing the violation type to it. Send() function will send the violated agent\_id and violation type to the server.
16. Provide a buffer time of 10sec to let the server update the database. After that call report() function of reporting\_agent class. report() function will print a notification message that will be sent to the user.
17. Return value of seat, agent\_id, reciever\_agent\_id and engine to agent class in form of object array list.
18. End.

**Algorithm for wrong direction class:-**

1. Begin
2. Set violation\_type as “Wrong direction”
3. Generate two random numbers in the range 1-10 for the violating agent coordinate and the reporting agent coordinate
4. Generate two random numbers (sno, rsno) in the range 1-10 to fetch the agent ids of the violating and the reporting agent such that the random numbers are not repeated
5. Fetch the agent ids of the violating and the reporting agent from the Reporting\_Agent class with the help of the random numbers generated (sno, rsno)
6. To check if there is any agent present in the area, the coordinates of the violating agent and the reporting agent are checked if they are equal. If false, value of flag is assigned to  zero.
7. If true :

* A random direction for the agent is generated
* Actual permitted direction for the area and the area in which the agent is present is fetched from the database using the gps class.
* The generated random direction is compared to the actual permitted direction, if false then the value of the flag is assigned to 1 and agent has to be reported  using the Reporting\_agent class with necessary details such as Agent ID of reporting and violating agent and the violation type. A buffer is added here so as to provide time for the server to fetch details from the database and update it accordingly.
* If the condition is false, that is the agent is moving in the permitted direction, the value of flag is assigned to 2.

1. An array list is returned to the Agent class with items such as the Agent ID of the reporting agent and the violating agent, value of flag to display appropriate statements in the Agent class and the name of the area in which the rule has been violated.
2. End

**Algorithm for speed limit class:-**

1. Begin
2. Set violation\_type as “Speed limit violation”
3. Generate two random numbers in the range 1-10 for the violating agent coordinate and the reporting agent coordinate
4. Generate two random numbers (sno, rsno) in the range 1-10 to fetch the agent ids of the violating and the reporting agent such that the random numbers are not repeated
5. Fetch the agent ids of the violating and the reporting agent from the Reporting\_Agent class with the help of the random numbers generated (sno, rsno)
6. To check if there is any agent present in the area, the coordinates of the violating agent and the reporting agent are checked
7. If true then, current speed of the agent is generated with the help of random numbers and the area of the agent present in using the gps class is fetched
8. Compare the current speed with the permitted speed of the area, if the speed is within the speed limit, there is no violation and that is displayed, else a buffer of 20 seconds is given for the agent to slow down
9. The new current speed is generated again with the help of random numbers and checked with the speed limit
10. If the new current speed is within the speed limit then no violation message is displayed, else the violation type and the details of the violation is sent to the reporting agent class
11. The details of the violation along with the fine is updated in the database
12. If the coordinates of the violating agent and the reporting agent are not equal then the message no agents detected will be displayed
13. Stop

**Algorithm for uninsured vehicle class:-**

1. Begin
2. Define initial value of check as 1.
3. Then generate two random serial numbers and call reporting agent class by creating its object and send two randomly generated serial numbers to that class.
4. Call violated\_agent and reciever\_agent function and get random violated and reporting agent id respectively.
5. Establish a connection with the database.
6. Fetch the expiry date of the insurance of the vehicle of the agent id generated from the database.
7. For current date, let date be 10 and month and year be random numbers. Month will be generated between 1 to 12 and year between 2018 to 2021.
8. Compare Expiry date and current date.
9. If the current date is greater than 0 then it means the insurance of the vehicle has expired already.
10. Give a warning to the user by sending notification to the user and provide a buffer of 3days to let the user update insurance of the vehicle. For simulation purposes, buffer can be taken as 10sec instead of 3 days.
11. Generate another random number to see if insurance has been updated or not after the warning. If the random number is equal to 1, then the insurance has been updated. If the random number is equal to 0, then it has not been updated even after the warning.
12. Search for a nearby agent and send agent id and violation type to the nearby agent through wifi.
13. If the current date is equal to 0 then it means that the insurance will expire today only.
14. Repeat steps 10 to 12.
15. If the current date is less than 0, then the Insurance is valid.
16. Return value of agent\_id, reciever\_agent\_id and check to agent class in form of object array list.
17. End.