QUESTION1: *Observe what you see with the agent's behavior as it takes random actions. Does the* ***smartcab*** *eventually make it to the destination? Are there any other interesting observations to note?*

After setting random actions for the primary agent, from the output below we can see that the smartcab reaches its destination time to time. It is reached frequently after the deadline.
We can observe that in some cases the reward is negative:
 - it is -1.0 when the smartcab makes an invalid move for the game
 - it is -0.5 when the smartcab does not take the optimal direction towards the target destination.

```
Simulator.run(): Trial 5
Environment.reset(): Trial set up with start = (1, 1), destination = (7, 5), deadline = 50
RoutePlanner.route_to(): destination = (7, 5)
LearningAgent.update(): deadline = 50, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = -0.5
LearningAgent.update(): deadline = 49, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = left, reward = -1.0
LearningAgent.update(): deadline = 48, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = -1.0
LearningAgent.update(): deadline = 47, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = -0.5
...
...
LearningAgent.update(): deadline = 4, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = forward, reward = -0.5
LearningAgent.update(): deadline = 3, inputs = {'light': 'red', 'oncoming': None, 'right': None, 'left': None}, action = None, reward = 0.0
LearningAgent.update(): deadline = 2, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = None, reward = 0.0
LearningAgent.update(): deadline = 1, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = None, reward = 0.0
Environment.act(): Primary agent has reached destination!
LearningAgent.update(): deadline = 0, inputs = {'light': 'green', 'oncoming': None, 'right': None, 'left': None}, action = right, reward = 12.0
```

QUESTION2: *What states have you identified that are appropriate for modeling the* ***smartcab*** *and environment? Why do you believe each of these states to be appropriate for this problem?*

The environment can be modelled with the following states:

- the traffic light *(inputs['light'])*: for the reward when going right on red and the penalty when going left or forward on red.
- the oncoming traffic *(inputs[‘oncoming’])*: for the agent being penalized when turning left on green light with oncoming traffic as this can cause an accident.
- the left value *(inputs[‘left'])*: is required when making a right turn on the red light, so an accident can be avoided .

The planner can be modelled with the state:

- next_waypoint *(next_waypoint())* which tells the desirable direction to take.

I choose to not include the right direction *(inputs[‘right’])* as the agent is not penalized for making a right turn on red because it is a legal move.

QUESTION3: *What changes do you notice in the agent's behavior when compared to the basic driving agent when random actions were always taken? Why is this behavior occurring?*

The driving agent can reach its destination remarkably faster than choosing random directions.
It has been developed for maximizing the reward at each state. This, for the first trips, is not always true, as the agent has a exploration_factor (epsilon) that allows taking into consideration also moves that consists in a negative reward. In this way, after the learning_factor overcomes the exploration_factor, the agent starts to be trained for optimal choices in the target direction without evading the rules of the game.

QUESTION4: *Report the different values for the parameters tuned in your basic implementation of Q-Learning. For which set of parameters does the agent perform best? How well does the final driving agent perform?*

For begin my tests i kept the exploration rate to 0.5 because I cannot predict its influence on reward and pass rate before evaluating learning rate and discount factor.

I start with this configuration

```
Learning Rate: 0.3, Discount Factor: 0.7, exploration_rate 0.5,
        Pass Rate: 97.0%, Explored States: 51, Mean Reward: 22.03, Number of Trials: 100

Learning Rate: 0.4, Discount Factor: 0.6, exploration_rate 0.5,
       Pass Rate: 99.0%, Explored States: 56, Mean Reward: 22.055, Number of Trials: 100

Learning Rate: 0.5, Discount Factor: 0.5, exploration_rate 0.5,
       Pass Rate: 98.0%, Explored States: 50, Mean Reward: 21.6, Number of Trials: 100

Learning Rate: 0.6, Discount Factor: 0.4, exploration_rate 0.5,
       Pass Rate: 99.0%, Explored States: 62, Mean Reward: 22.185, Number of Trials: 100

Learning Rate: 0.7, Discount Factor: 0.3, exploration_rate 0.5,
       Pass Rate: 100.0%, Explored States: 59, Mean Reward: 22.2, Number of Trials: 100
```

I saw that as we increase the learning rate, that the mean reward and the passing rate has averagely increased. A second try confirmed it.
However with `Learning Rate: 0.7, Discount Factor: 0.3` the results where somewhat unstable so I considered for testing the exploration rate this values: `Learning Rate: 0.6, Discount Factor: 0.4`

```
Learning Rate: 0.6, Discount Factor: 0.4, exploration_rate 0.5,
       Pass Rate: 100.0%, Explored States: 56, Mean Reward: 22.66, Number of Trials: 100

Learning Rate: 0.6, Discount Factor: 0.4, exploration_rate 0.6,
       Pass Rate: 98.0%, Explored States: 55, Mean Reward: 22.22, Number of Trials: 100

Learning Rate: 0.6, Discount Factor: 0.4, exploration_rate 0.7,
       Pass Rate: 99.0%, Explored States: 62, Mean Reward: 22.925, Number of Trials: 100

Learning Rate: 0.6, Discount Factor: 0.4, exploration_rate 0.8,
       Pass Rate: 99.0%, Explored States: 49, Mean Reward: 22.45, Number of Trials: 100
```

Choosing slightly higher values exploration rates consisted in an higher number of explored states and thus in an higher mean reward.

For the optimal settings i choose:

```
Learning Rate: 0.6, Discount Factor: 0.4, exploration_rate 0.7,
```

QUESTION5: *Does your agent get close to finding an optimal policy, i.e. reach the destination in the minimum possible time, and not incur any penalties? How would you describe an optimal policy for this problem?*

It is really important in our scenario to evaluate what criteria we want to give precedence to:

- mean reward → this could be the optimal criteria for example in a taxi-driver scenario
- pass rate → this would be the optimal criteria for example in driving licence exam.

It is also important to consider that the deadline random parameter influences the passing rate which is, in a reinforcement learning problem, the parameter that gives the real randomness to our world.

For example an optimal policy could decide to skip a trip if, at the beginning, we know the deadline is too close for risking to loose reward on this trip.