QUESTION: *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?*

> The cab moves around randomly as expected, occasionally reaching the destination but more often not. Out of 100 trials, it reaches the destination 10 times. It often attempts to go forward or left when at a red light and thus stays in place. The borders are not hard borders but rather transports the agent to the opposite edge.

QUESTION*: 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 state takes light, left, oncoming from inputs as well as next_waypoint. These are appropriate because it supplies the information about what the smartcab can do at the intersection, and where the smartcab wants to go next. Right can be ignored because it has no relevance to what action the smartcab can take (assuming that it obeys traffic laws). Deadline is also ignored because including it would drastically increase the number of states and make it so that the agent doesn't learn effectively.

OPTIONAL: *How many states in total exist for the* ***smartcab*** *in this environment? Does this number seem reasonable given that the goal of Q-Learning is to learn and make informed decisions about each state? Why or why not?*

How many total states can be determined by looking at each state variable:

- Light: Green, Red →2
- Left: forward, right, left, none →4
- Oncoming: forward, right, left, none →4
- Next_waypoint: forward, right, left →3

Total States = 96. This seems like a reasonable number of states as we attempt to run with a large number of trials, especially given the fact that there is limited traffic so many of the states with traffic from left or oncoming will not be reached

QUESTION: *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 agent begins to learn during the first trial, often reaching the destination on the first trial (though after taking some incorrect actions and often crashing!!!) and steady improves as more trials are run. This is because the actions and rewards or each state are saved in the qtable and called via the qlearning algorithim. As more state, action pairs gain values, and as those values get updated to increase accuracy, the agent becomes better.

QUESTION: *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?*

Alpha	gamma	Epsilon	Successes	Penalties
0	1	0	1	100
1	1	1	10	100
0	0	1	17	100
1	1	0.5	28	96
1	0	0.5	73	69
0.6	0.4	0.1	90	70
0.9	0.8	0	92	48
0.2	0.4	0.1	96	68
0.6	0	0.1	98	58
1	0	0	99	20
0.9	0	0	99	18

The agent performed best with a high alpha value and a low gamma and epsilon value which makes sense because that will put the most weight in the immediate reward. By just focusing on the immediate reward the agent, after some learning the agent will always reach the destination with minimal errors. The final driving agent while using .9 as alpha and 0 as gamma and epsilon will always reach the destination after the first one or two trials. Having a higher gamma will make sense in different training scenarios where total utility is less reliance on the immediate reward having a higher epsilon will make sense when there are more actions that give positive rewards and there is a chance you find and get stuck on a decent action but not the best action

QUESTION: *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?*

The agent definitely gets close to finding the optimal policy – after sufficient tuning it will always reach the destination without penalty. The question is whether it reaches in minimum time and that I am not sure of, for instance there are circumstances where the next way_point is forward while going right could be just as good in the long term, and if it's a red light then going right might actually be better. The optimal policy would seem to be going to the next way_point if not breaking any traffic laws and mostly staying in place if traffic doesn't allow it to go to the next way point while

sometimes going right on red even if next_waypoint is forward depending on where the ultimate destination is.