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?

There aren't many observations to notice, except for the fact that the smartcab drives in circles. I've run the test 5 times for 1000 trials each and got the following results:

Trials	Success	Ran Out of time	Hard Deadline
1. 1000	682	475	318
2. 1000	666	460	334
3. 1000	655	474	345
4. 1000	678	476	322
5. 1000	652	426	348

Success rate if we don't enforce the deadline (<= 100 moves) - 66.66%, if we enforce the deadline it is 20.44%.

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?

I decided to include following states: light (green, red), oncoming (None, left, right, forward), left (None, left, right, forward) and a next waypoint.
We are using here the US driving laws, which means that traffic to the right don't affect our behaviour on a light-regulated intersection.
Next waypoint is our goal and we need to understand can we proceed there or not.
Only the light, oncoming and left traffic can help us to understand what is going to be our next move and whether it is going to be an allowed move.

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?

Smartcab started reaching destination each time while not hitting a hard-deadline.
It improves with each run and consistently achieves 80%+ success rate. It does so, because it uses a Q learning algorithm instead of taking random actions.

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?

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?

I've tried following sets of parameters:
alphas = [1, 0.8, 0.6, 0.3, 0.1]
gammas = [0.5, 0.3, 0.1]
epsilons = [0.2, 0.1, 0.05, 0.01, 0.005]

And for parameters Alpha 0.8, Gamma 0.1, Epsilon 0.005 it consistently performs with 99% success rate: 13.7 moves on average, 23 rewards average, 0.6 penalties average.

Optimal solution is to reach the destination on time and taking as less as possible moves, while getting minimum penalties.