Questions:

Implement a Basic Driving Agent
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?

Most of the time it eventually reaches its destination, though sometimes it hits the hard deadline of 100 moves plus the original deadline. I noticed that rewards are always 0 for not moving. The rewards were all in the set {12, 2, 0, -0.5, -1}. 12 was awarded for reaching the destination. All the -1 rewards occur with red lights.

Though the agent does make it to its destination eventually, it incurs negative rewards throughout the process as it isn't learning to avoid them.

Also, I noticed that traffic was uncommon.

Inform the Driving Agent
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 chose light, which stores the state of the light, as reward numbers depend it. The same goes for traffic heading toward the intersection in the forward and left directions in the previous state. Finally, I chose the next waypoint, as rewards should eventually depend on whether the cab is going the right way toward the goal. I ignored location, as the agent doesn't really need this information if it already knows which direction leads to the goal, and it would greatly increase the number of states to use it.

The time and deadline values are also available, but as they would very greatly increase the number of states to explore, to a point that makes the algorithm not practical, I choose to leave them out. I also left out traffic for the right direction, as, according to US traffic rules, whether or not traffic is there will not affect the reward, as it would be stopped if the light green, and if it is red only a right turn is allowed (and wouldn't come into contact from cars on the right, as they are on the opposite side of the road).

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?

Number of possible light states per intersection * number of oncoming traffic values (which would be 3^3) per intersection * number of possible waypoints (3) = 2*27*3 = 162
.

This number of states requires quite a few moves to visit all of the states, but over several trials it is reasonable to assume the agent will visit a significant number of the states.

Implement a Q-Learning Driving Agent

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?

In the first trials the cab took longer to get to the goal. After a few trials it nearly always got to the goal by the deadline. This is because, after the first trial, the reward of reaching the goal starts to be reflected in actions that lead toward the goal. It also fairly quickly learns to stop at lights (or make a right turn if the goal is in that direction).

Improve the Q-Learning Driving Agent

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?

My tuning efforts are displayed below.

alpha	gamma	epsilon	% Success	Average reward over actions
.5	1/(t+1)	.2	92	1.41
.9	same	same	93	1.30
.1	same	same	95	1.40
.5	same	.2 - .002 * t	93	1.40
.5	same	.1 - .001 * t	99	1.62
.5	.5	same	98	1.63
.5	1	same	100	1.64

The algorithm performed best in the the last row in terms of accuracy and its average reward. The algorithm performs best when the update and original reward are weighted the same, epsilon is small and decaying (random actions are chosen infrequently at the beginning and less frequently as trials continue), and emphasis is on future rewards with a gamma of 1.

The final driving agent performs mostly well after it has run sufficient trials, though it does occasionally receives negative rewards.

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 think it gets fairly close, as negative rewards are only in a small minority of actions, and watching the car on the board reveals that it moves very directly toward the goal. An optimal policy would take the directest route to the goal that incurs no negative rewards.