**Linear problem – mathematical questions:**

1. Assuming an optimal, completely greedy policy, and a discount factor of gamma = 0.9, calculate the Q-value of each (state, action) pair.

2. Under the same assumptions, calculate the value of every state (this shouldn't be much work given the last part).

**Linear problem – code questions:**

3. About how long (how many training episodes) does it take the tabular Q-system to converge to the optimal Q values you calculated above?

4. For which states do the Q-values converge earlier? For which actions? Why?

5. How does changing epsilon affect this?

**Cartpole problem - conceptual questions:**

7. Since the reward for every episode (not every action!) will be -1, why would a Q-learning system learn any interesting behavior on this task?

8. Why might a DQN (or some other function approximator) be an appropriate choice here? Compare this with the linear track problem, would a DQN be helpful there, and if so, is it for the same reason as here?

**Cartpole problem – tabular controller questions:**

9. The tabular Q-learning system does much better than a random controller, but it still only lives about 5 times as long. What could we do to improve the tabular Q system's performance on this task further? For whatever you propose, how would it affect training?

10. Try setting gamma = 0.0 (living in the moment). What happens? Why?

11. What happens if we set gamma = 1 (living in all moments at once)? Naively, one might expect to get random behavior, since all trials get the same total reward, and gamma = 1 is essentially saying that the total reward is all that matters, not when the reward appears. However, this is not what actually happens. Why?

12. What happens if you set epsilon = 1 (random behavior while training)? Why?

13. What happens if you set epsilon = 0 (no exploration)? Why does this happen here, and what might be different about other tasks that makes exploration important?

**Cartpole problem – DQN controller questions:**

14. Why does the DQN take more episodes to train than the tabular Q-learning system?

15. In my implementation, I used the tanh activation function at the output layer. Why might this be an appropriate choice here? More specifically, what are some activation functions that would probably NOT yield good results at the output layer?

16. What happens if we turn off the replay buffer? Why might it be important?