Reinforcement Learning Mini Project

In this mini project, you will implement a reinforcement learning agent to play a game of your choice.

Deliverables

You should hand in a `.zip` file with the following contents:

1. A `.py` script that I can run to see your agent play the game.
2. A `.md` file with 300-ish words that explains your implementation and the results of your experiments.
3. A `.pkl` file with the trained parameters of your agent.
4. A `.gif` of your agent playing the game.
5. A `.gif` of a random agent playing the game.

You can use any reinforcement learning algorithm you like, but I recommend using DQN.

Submission

Send the `.zip` file to nobr@itu.dk. The deadline is 23:59 on October 1st.

Constraints

You are allowed to use `random, tree, grad, jit, lax` and `vmap` from `jax` as well as `optax`, and `chex` (and of course `gymnax`), but no other deep learning libraries (so also not `jax.nn`). You could use `lax.scan` or a simple for loop to play the game, while storing the transitions in a buffer. The buffer should be a `deque` from the `collections` module (store `n` entries and throw away the oldest when beyond capacity).

Some kind words

Deep Q-learning is a simple and powerful algorithm, but it can be a bit tricky to get right. From a signal processing perspective, it is actually an *infinite impulse response filter*, and there is a recurrent aspect to it that can be a bit tricky to wrap your head around.

Talk to each other, ask questions, make sure your environment is set up correctly.

Bonus

Change out your environment for a different one, to confirm the generality of your implementation.