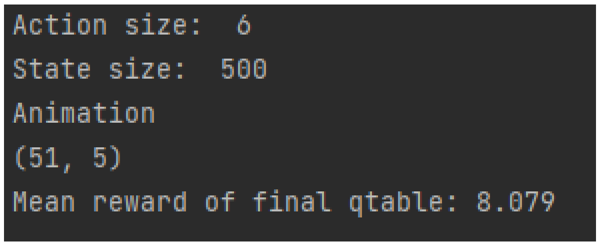
Exercise Set 5: Reinforcement learning Valtteri Nikkanen

1. Implementing Q-learning to solve the Taxi problem we can achieve a good/"optimal" policy with about 5000 episodes, with the chosen parameters. In figure 1 the code prints are presented and in figure 2 the graph displaying the evolution of the policy over the episodes.



```
Action size:  6
State size:  500
Animation
(51, 5)
Mean reward of final qtable: 8.079
```

Figure 1: Code prints for Q-learning

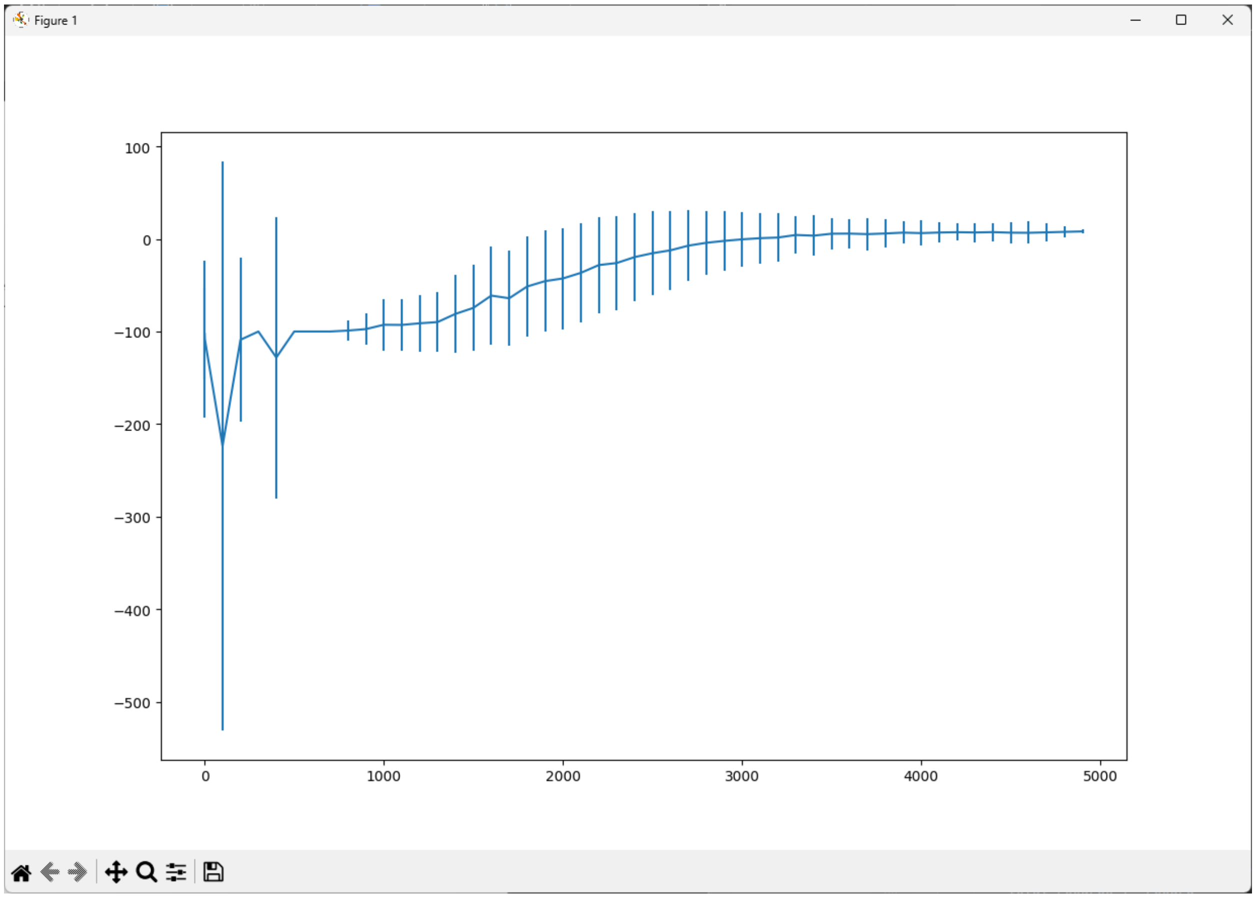
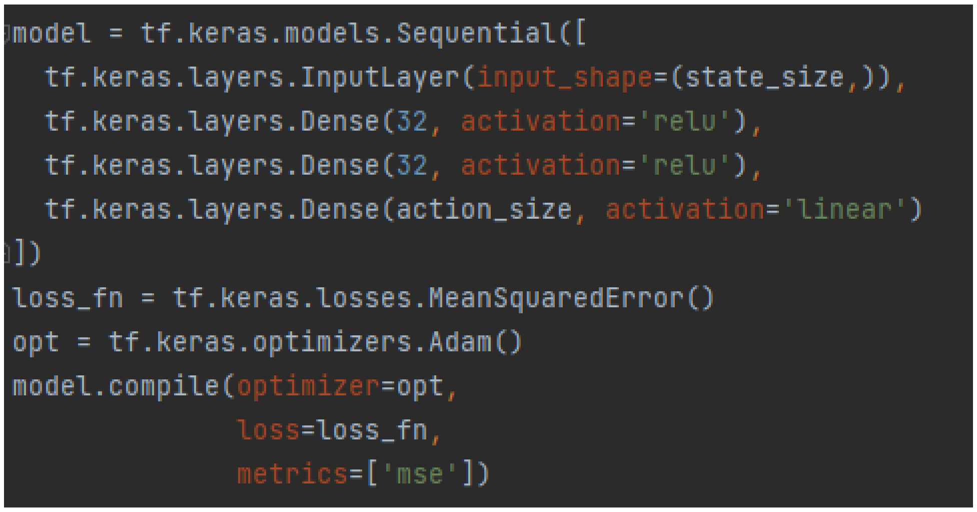


Figure 2: Evolution of the mean reward with its standard deviation

The mean reward goes to around 7-8 depending on the starting position of the taxi, passenger and goal.

The Q-table is then saved to a file so it can be used for the neural network. There is also an animation of the found path by the Q-learning.

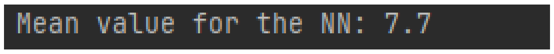
2. We save the Q-table from the first code to a file and use it to train the neural network. I chose to have a simple fully connected neural network which is defined as shown in figure 3.



```
model = tf.keras.models.Sequential([
  tf.keras.layers.InputLayer(input_shape=(state_size,)),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dense(action_size, activation='linear')
])
loss_fn = tf.keras.losses.MeanSquaredError()
opt = tf.keras.optimizers.Adam()
model.compile(optimizer=opt,
              loss=loss_fn,
              metrics=['mse'])
```

Figure 3: Defining the neural network

The network is then fed the Q-table and it is trained for 200 epochs. 100 was too few as the nn didn't manage to learn what to do in that time. After that the code tells the mean value of the neural networks result, which is shown in figure 4.



```
Mean value for the NN: 7.7
```

Figure 4: Neural network results.

After this an animation is played about the path the neural net creates.