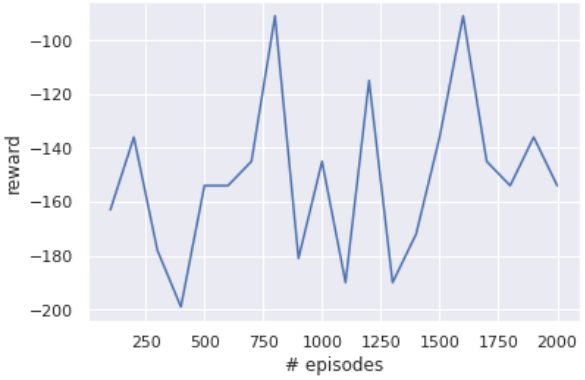
Q-Learning (ALPHA=0.1 EP=2000)

And the rewards

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



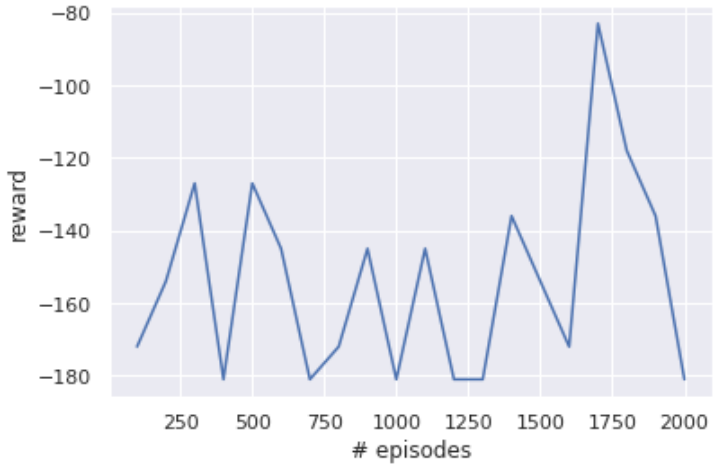
SARSA(ALPHA=0.1 EP=2000)

And the rewards

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```

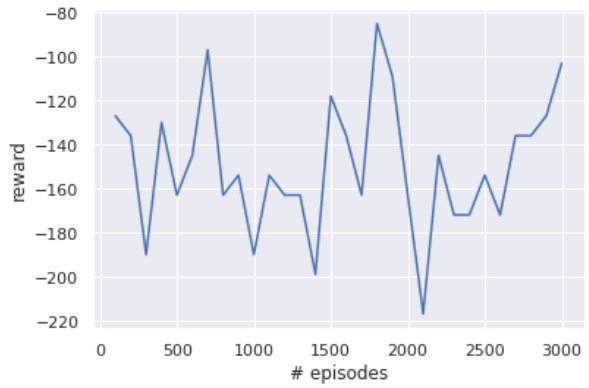


Q-Learning(ALPHA = 0.3 EP = 3000)

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



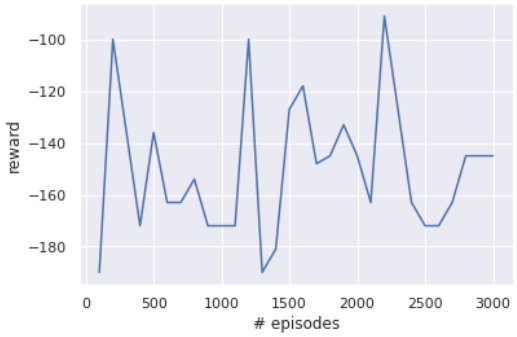
4.2 SARSA (25p)

SARSA(ALPHA = 0.3 EP =3000)

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



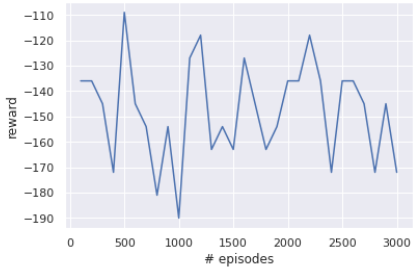
Q-LEARNING(ALPHA=0.3 GAMMA=0.5 EP = 3000)

And the rewards

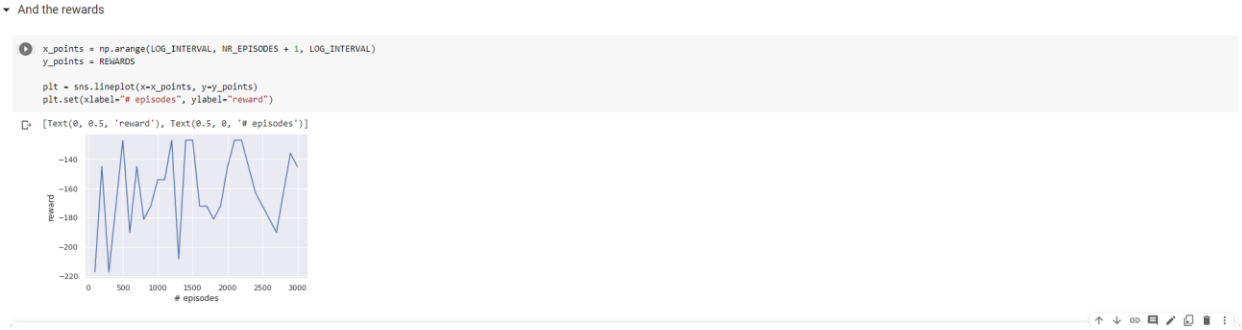
```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



SARSA(ALPHA=0.3 GAMMA=0.5 EP = 3000)



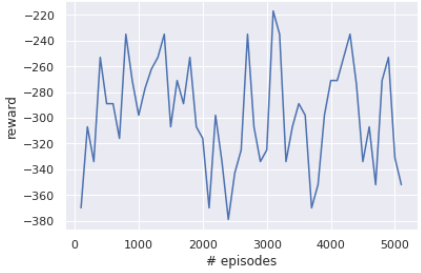
Q-Learning(ALPHA=0.5 GAMMA=0.5 EP=5100)

And the rewards

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



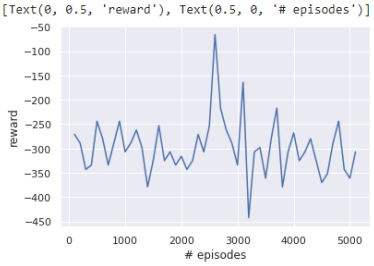
SARSA(ALPHA=0.5 GAMMA=0.5 EP=5100)

And the rewards

```
[23] x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
     y_points = REWARDS

     plt = sns.lineplot(x=x_points, y=y_points)
     plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



4.3 Results (35p)

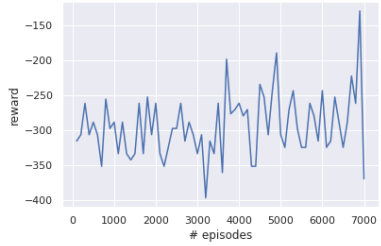
Q-Learning(ALPHA =0.4 GAMMA=0.7 EP=7000)

And the rewards

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



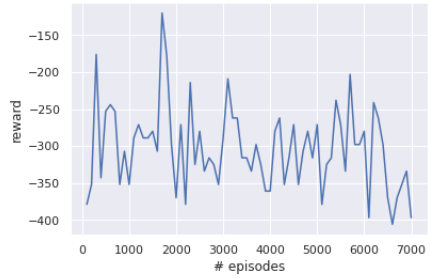
SARSA(ALPHA =0.4 GAMMA=0.7 EP=7000)

And the rewards

```
x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
y_points = REWARDS

plt = sns.lineplot(x=x_points, y=y_points)
plt.set(xlabel="# episodes", ylabel="reward")
```

```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



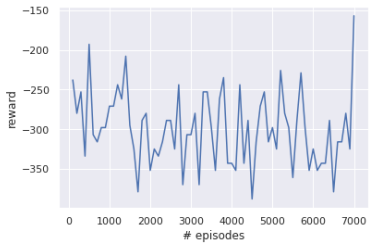
Q-Learning(ALPHA = 0.3 GAMMA =0.8 EP=7000 DECAY_EPS=0.85)

And the rewards

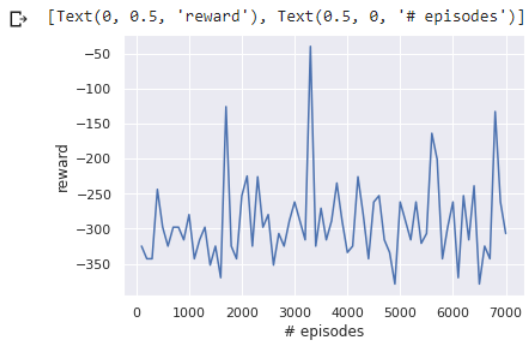
```
[17] x_points = np.arange(LOG_INTERVAL, NR_EPISODES + 1, LOG_INTERVAL)
     y_points = REWARDS

     plt = sns.lineplot(x=x_points, y=y_points)
     plt.set(xlabel="# episodes", ylabel="reward")
```

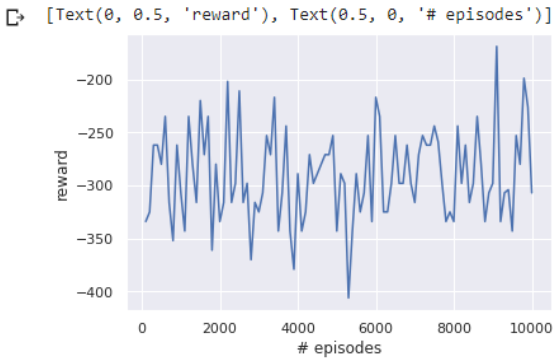
```
[Text(0, 0.5, 'reward'), Text(0.5, 0, '# episodes')]
```



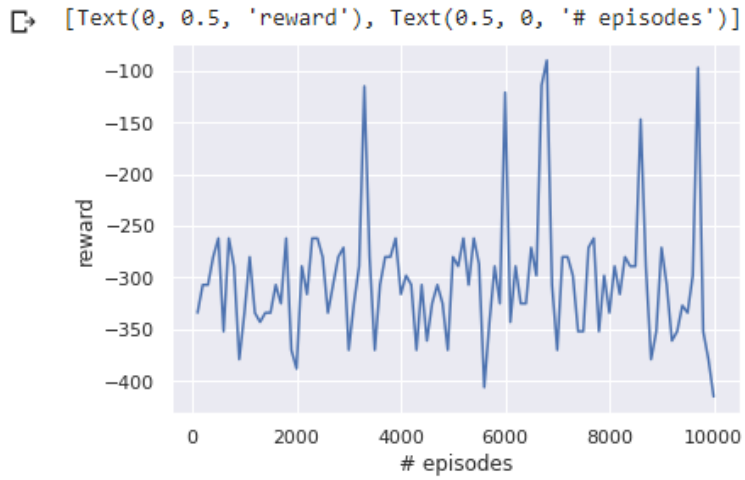
SARSA(ALPHA = 0.3 GAMMA =0.8 EP=7000 DECAY_EPS=0.85)



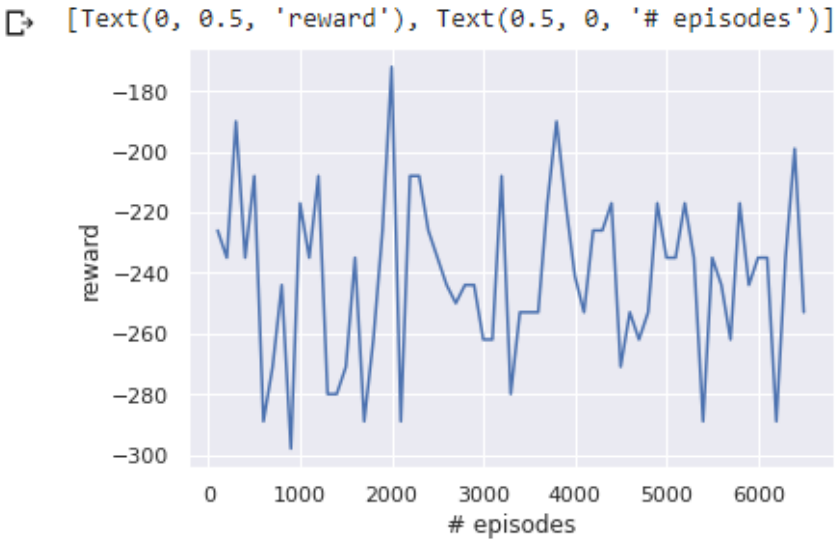
Q-Learning(ALPHA = 0.3 GAMMA =0.8 EP=10000 DECAY_EPS=0.75 EPSILON=0.5)



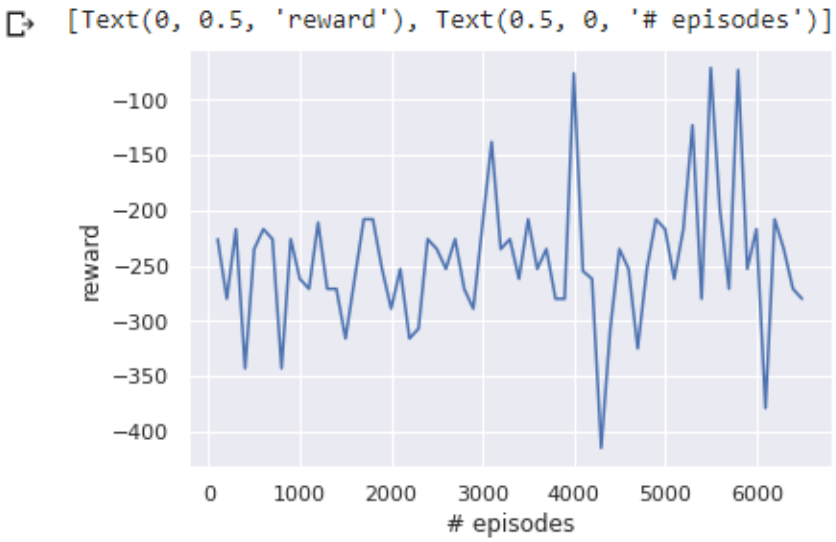
SARSA(ALPHA = 0.3 GAMMA =0.8 EP=10000 DECAY_EPS=0.75 EPSILON=0.5)



Q-Learning(ALPHA = 0.8 GAMMA=0.4 EP = 6500)



SARSA(ALPHA = 0.8 GAMMA=0.4 EP = 6500)



Analiza:

Cu cat algoritmii sunt rulati pe mai multe episoade,cu atat algoritmii invata mai mult,dar nu inseamna ca mai bine sau eficient in mod neaparat.

Daca learning rate-ul este mare(alpha) atunci ar aparea o problema sezibabila in invatare,si s-ar observa si pe grafice,dar in cazul nostru prea este posibil deoarece avem acea “atenuare” facand (1-alpha).

Implementare:

Implementarea a constat in crearea unei functii choose_action care intoarce o actiune cu o probabilitate eps,altfel se maximizeaza utilitatea in actiunea curenta.

- Q-Learning: primul pas este alegerea unei actiuni,se updateaza starea curenta cu noua valoare dupa formula predate la curs si se trece la new state.
- SARSA: primul pas este alegerea unei actiuni,se updateaza starea curenta cu noua valoare dupa formula predate la curs si se trece la new state.Diferit fata de Q-Learning este faptul ca aici ne folosim de starea urmatoare atunci cand calculam valoare starii curente.