Міністерство освіти і науки України

Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Програмування інтелектуальних інформаційних систем

**ЗВІТ**

до лабораторних робіт

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Виконав**  **студент** |  | ІТ-92 Щур А. С. |  |  |
|  |  | (№ групи, прізвище, ім’я, по батькові ) |  |  |
|  |  |  |  |  |
| **Прийняв** |  | пос. Баришич Л. М. |  |  |
|  |  | (посада, прізвище, ім’я, по батькові ) |  |  |

Київ 2021

# **Завдання лабораторної роботи 6**

# Завданням даної роботи є застосувати бібліотеку DQN до вашої гри. В результаті, ваша гра має грати сама у себе - проте поведінкові зміни відбуватимуться саме на основі візуальної зміни стану на екрані. Задача про "гру в саму себе" подібна до Lab03, де ви застосовували Minimax алгоритм, щоб гравець пересувався полем. Тут мета і факт гри є ідентичним.

# Процес навчання та прийняття рішень необхідно записувати в окремий файл або логувати в консолі. Ви можете також застосувати альтернативний підхід до відображення результату - наприклад, візуалізацію.

# **Опис програмного коду**

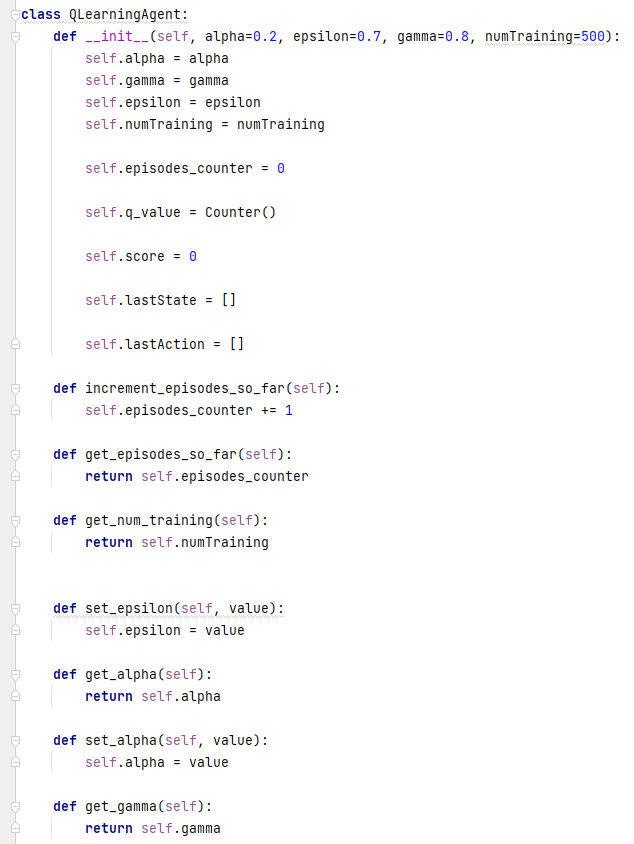
Я використовував чистий Q-learning, суть даного методу, що у нас є умовно таблиця станів нашої системи і рухів, які ми можемо зробити в поточному стіні цієї системи. І оскільки, спочатку агент не знає коефіцієнтів, тобто таблиця не заповнена, він має ходити випадково, таким чином досліджуючи гру. Після дослідження системи, та після заповнення нашої матриці станів, гравець починає використовувати її для досягнення бажаного результату.

Я додав до свого проекту клас Q-агента, при ініціалізації ми вказуємо йому такі важливі параметри як Alpha Gamma Epsilon, так к-сть циклів навчань.

Alpha відповідає за швидкість навчання

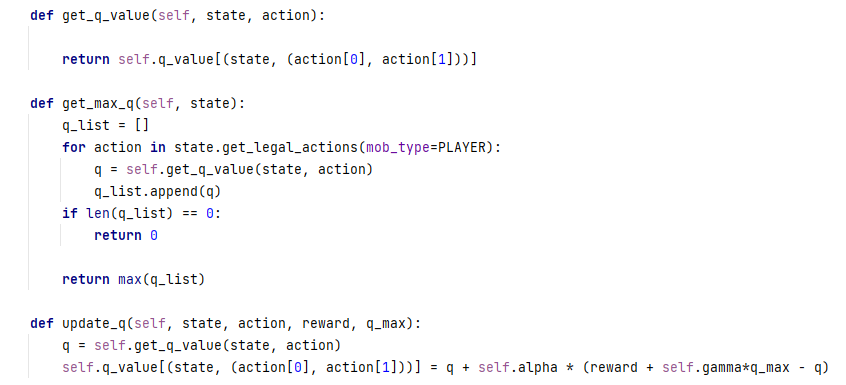
Gamma відповідає за вибір оптимального ходу якщо фінальна ціль близько

Epsilon відповідає за те, чи буде ходити агент випадково, чи буде використовувати вже набуті знання

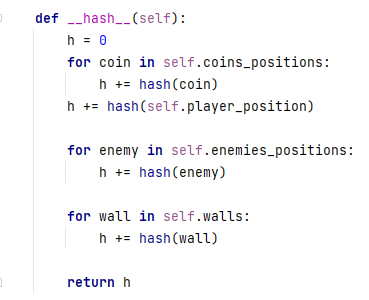


Далі подивимось на те, як ми будемо записувати ціну ходу із стану.

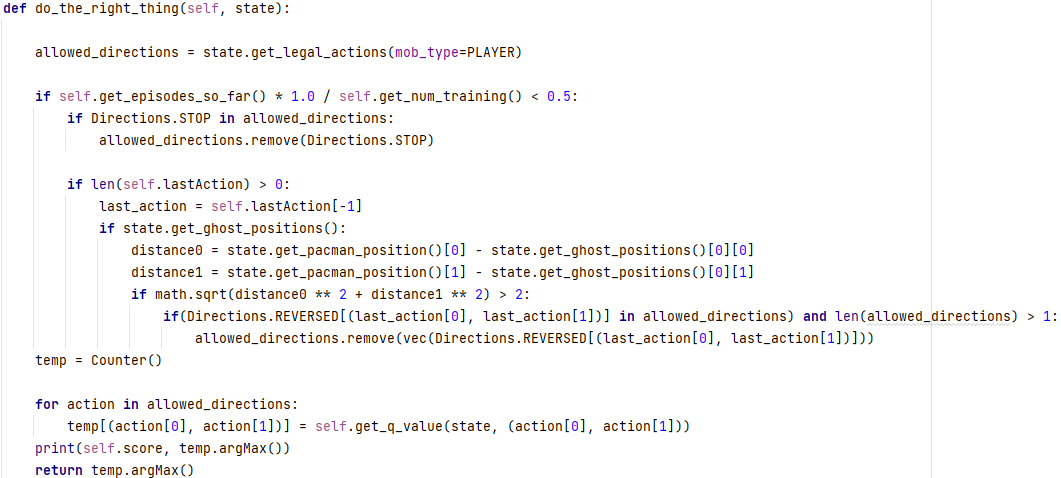
Я використовував звичайний словник, трохи модифікований. Для стану я перевизначив метод хешування.



Метод хешування для стану

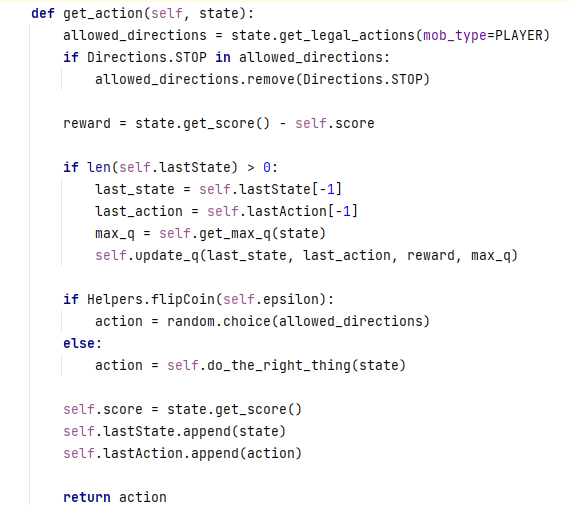


Далі подивимось на головні функції, що допомагають агенту навчатись та обирати хід

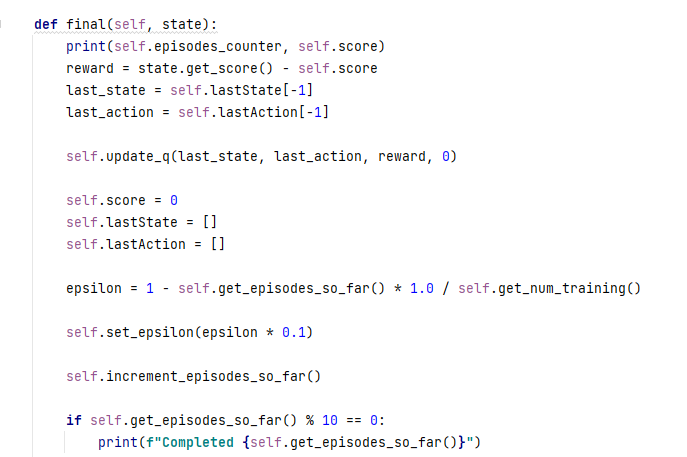


В загальному в даному методі шукається оптимальний хід зі словника знань, але якщо агент мало повчився, то я йому трошки допомагаю і він тікає від ворога(точніше не йде в його сторону) якщо ворог знаходиться в радіусі однієї клітинки.

Далі подивимось на те, як агент навчається



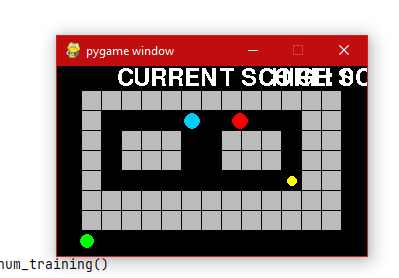
Ми обчислюємо Q-значення в кожному стані, потім підкидуємо монетку, яка говорить нам, використовувати набуті знання чи продовжувати вивчати систему і робити хід випадкого(тут слід наголосити, що це залежить від параметра Epsilon, чим він менший, тим менше агент ходить випадково).



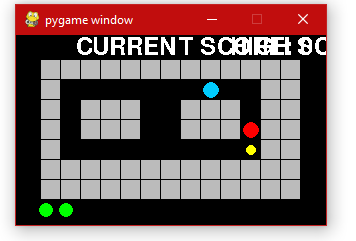
Фінальний метод викликається при перемозі чи поразці, де йде обнулення нагород(але ми не обнулюємо словник знань). Та є вивід в консоль пари значень

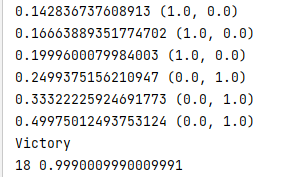
**Робота гри**

Процес навчання Пакмана



Пакман їсть монетку і перемагає





**Висновок**

В даній роботі я реалізував Q-навчання для мого агента, і досить добре зрозумів саму концепцію навчання з підкріпленням