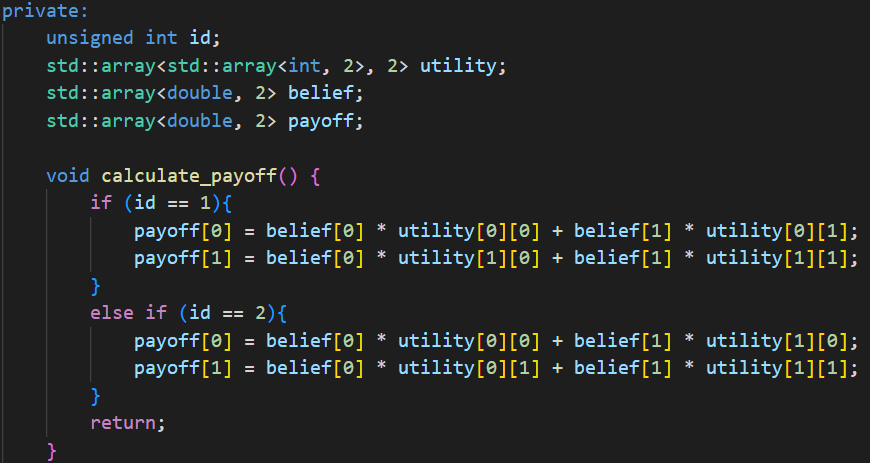
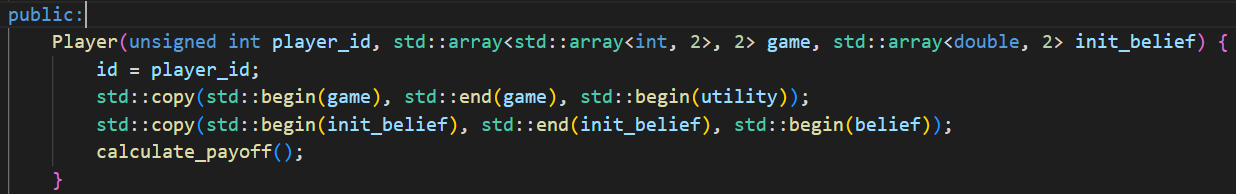
**Game Theory Homework 2 Report**

**311551094 資科工碩一 廖昱瑋**

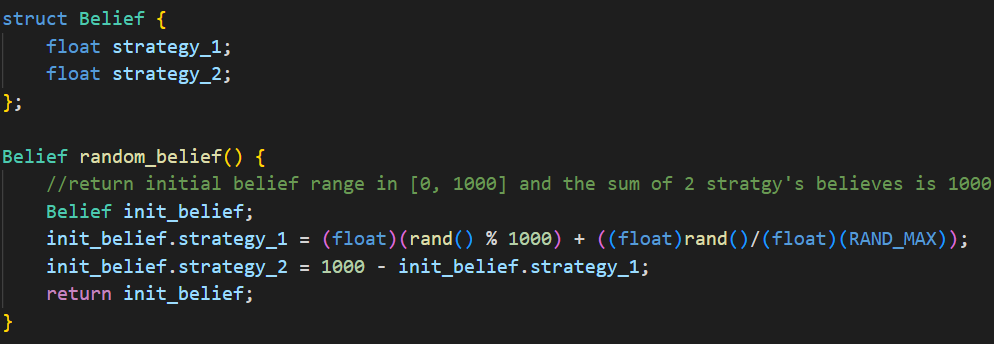
1. Code

定義一個class名為 Player，裡面變數有player的id、他個人的utility matrix、他對於對手的belief及他的payoff。其中，於*calculate\_payoff*函數計算payoff。

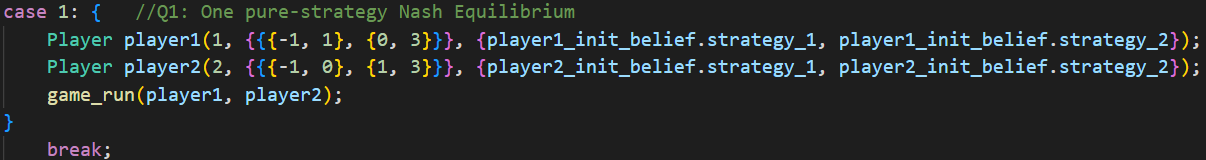




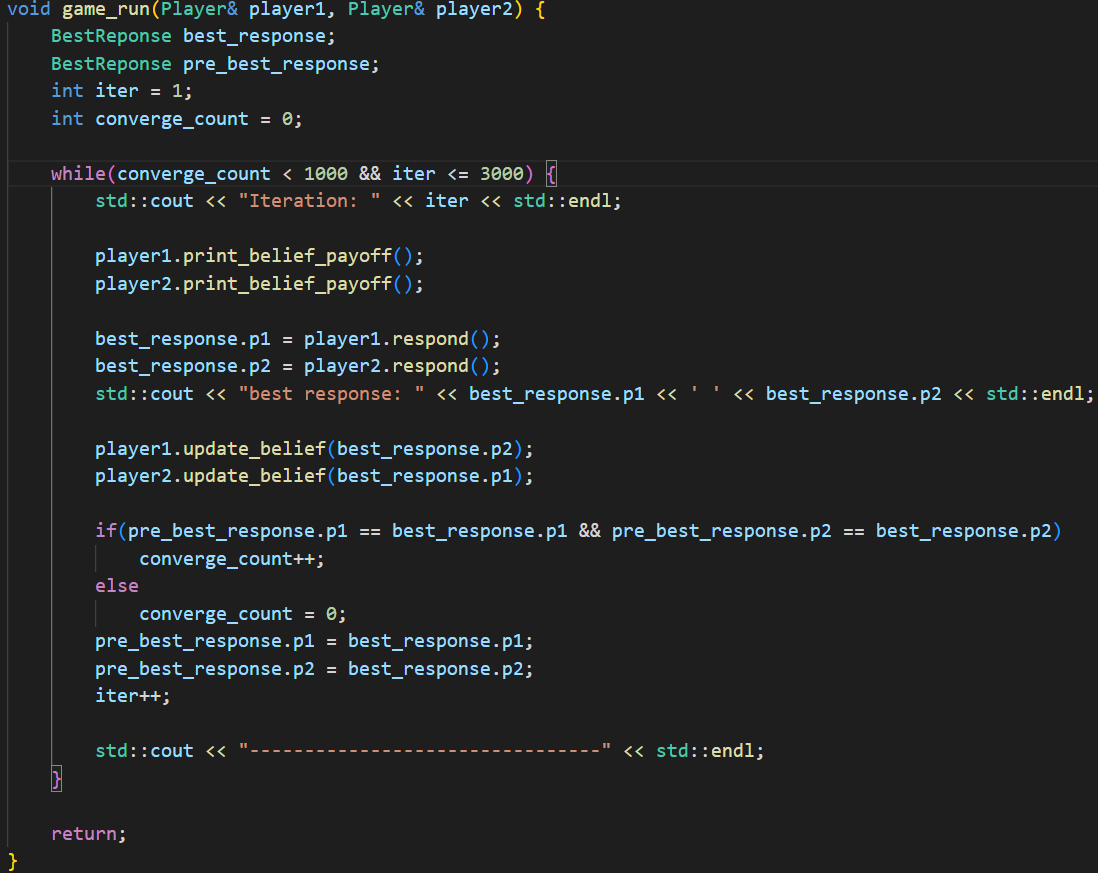
初始的belief以pseudo random function產生，player中兩個strategies的belief加總為1000，且型別為float。



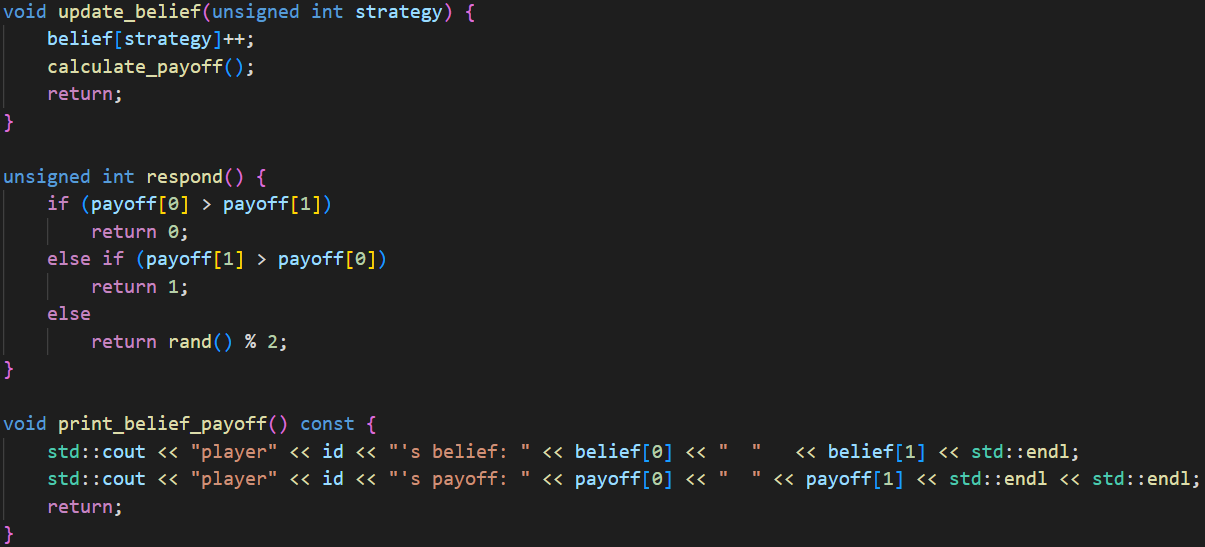
這次作業總共有9小題，程式中一開始會先請使用者輸入想解哪一題。為了方便解釋，以這邊以Q1為例，其他questions只是輸入的utility matrix不同。先宣告兩個Player object player1及player2，參數依序為id、utility matrix及初始belief。接著進行game。



*game\_run*函數中，會先print出兩個players當前的belief及payoff，再以payoff決定這輪的best response，最後用對手的best response更新自己的belief及payoff。不斷的重複做迴圈，收斂條件為兩個players的best response 1000回合都沒改變或該game進行超過3000回合。

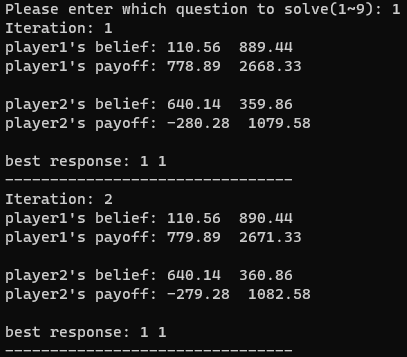
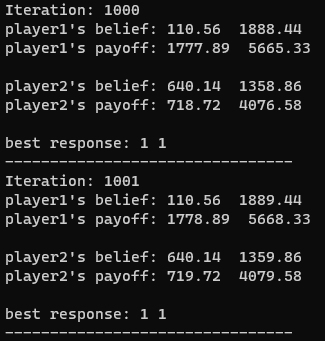


*update\_belief*、*respond*及*print\_belief\_payoff*函數定義如下，值得注意的是在*respond*中，若player的兩個payoff相同，就隨機選擇best response；以及在*update\_belief*中，除了更新belief，還會更新計算payoff。



1. Questions
   * 1. One Pure-Strategy Nash Equilibrium

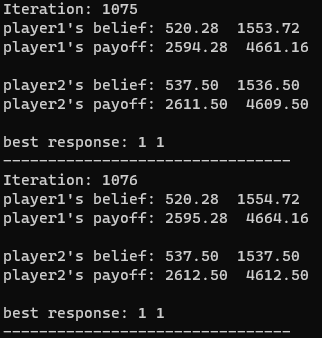
這個game會收斂到pure-strategy NE (r2, c2)。設player1’s belief (a, b)、player2’s belief (c, d)，則player1’s payoff為 (-a+b, 3b)、player2’s payoff為 (-c+d, 3d)，因a, b, c, d皆為非負的float，兩players的best response永遠都是(r2, c2)。下圖為程式執行範例，左圖為起始點，右圖為結束點。

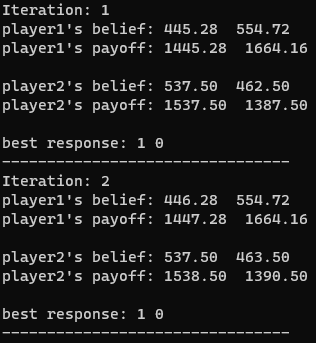
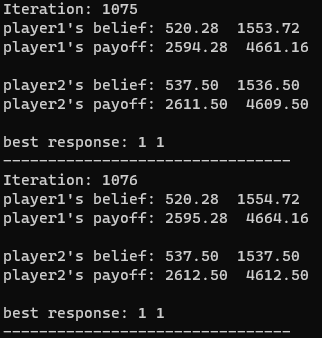
* + 1. Two or More Pure-Strategy NE

這個game有可能收斂到兩個pure-strategy NE (r1, c1)、(r2, c2)，也有可能在(r1, c2)、(r2, c1)之間一直跳，收斂到mixed-strategy NE，、、、。

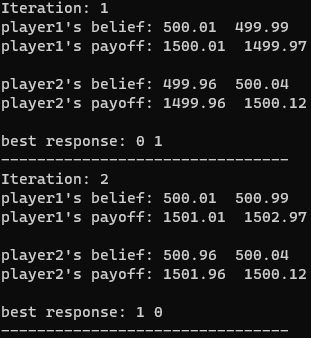
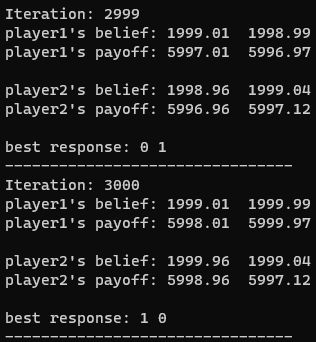
* + - 1. 收斂到pure-strategy NE (r1, c1)，自從game進入到(r1, c1)後，兩個players的payoff變動皆為(+2, 0)，這會使得接下來所有回合都選擇(r1, c1)。

* + - 1. 收斂到pure-strategy NE (r2, c2)，自從game進入到(r2, c2)後，兩個players的payoff變動皆為(+1, +3)，這使得接下來所有回合都選擇(r2, c2)。

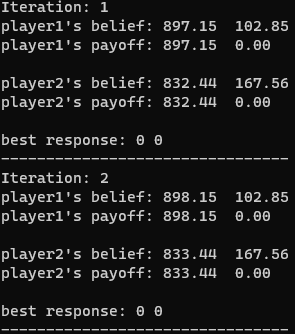
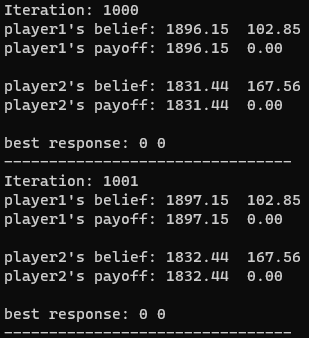
 

* + - 1. 收斂到mixed-strategy NE，如果起始belief接近、、、，但有些微不同，則會造成結果於(r1, c2)、(r2, c1)之間一直跳。因為各player之內的初始payoff差值小於1，在best response為(r1, c2)、(r2, c1)狀況下，都是造成比較小的payoff再加1，這樣上一輪payoff較大的一方在下一輪會變成比較小，如此不斷循環，最後結果為mixed-strategy NE。

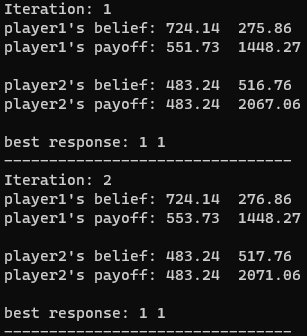
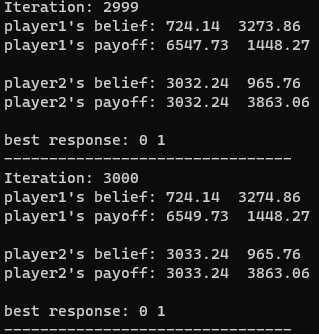
* + 1. Two or More Pure-Strategy NE (Conti.)

這個game會收斂到pure-strategy NE (r1, c1)。設player1’s belief (a, b)、player2’s belief (c, d)，則player1’s payoff為 (a, 0)、player2’s payoff為 (c, 0)，因a, c為非負的float，players的best response會很快的收斂到(r1, c1)。

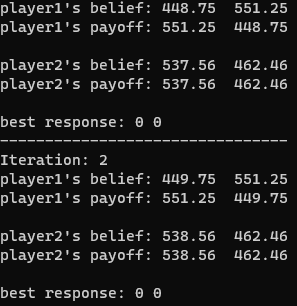
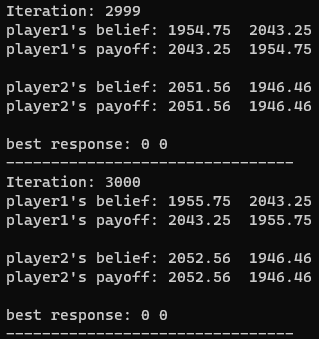
* + 1. Mixed-Strategy Nash Equilibrium

這個game會收斂到mixed-strategy NE，、、、。Best response為(r1, c2)時，player1的payoff變動為(+2, 0)、player2的payoff變動為(+1, 0)，直至player2 best response變為c1。在best response為(r1, c1)，player1的payoff變動為(0, +2)、player2的payoff變動為(+1, 0)，直至player1 best response變為r2。在best response為(r2, c1)， player1的payoff變動為(0, +2)、player2的payoff變動為(0, +4)，直至player2 best response變為c2。在best response為(r2, c2)，player1的payoff變動為(+2, 0)、player2的payoff變動為(0, +4)，直至player1 best response變為r1。所以又回到了(r1, c2)，進入無限輪迴。依據以上四種best response下payoff的變動程度，可得best response機率為、、、。

* + 1. Best-Reply Path

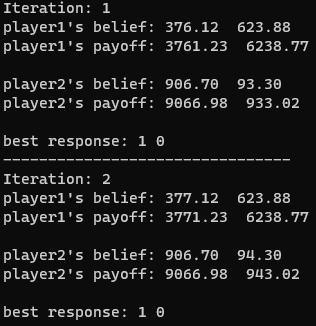
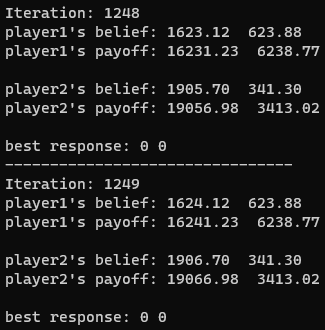
這個game會收斂到mixed-strategy NE，、、、。Best response為(r1, c1)時，player1的payoff變動為(0, +1)、player2的payoff變動為(+1, 0)，直至player1 best response變為r2。在best response為(r2, c1)，player1的payoff變動為(0, +1)、player2的payoff變動為(0, +1)，直至player2 best response變為c2。在best response為(r2, c2)，player1的payoff變動為(+1, 0)、player2的payoff變動為(0, +1)，直至player1 best response變為r1。在best response為(r1, c2)，player1的payoff變動為(+1, 0)、player2的payoff變動為(+1, 0)，直至player2 best response變為c1。所以又回到了(r1, c1)，進入無限輪迴。依據以上四種best response下payoff的變動程度，可得best response機率為、、、。

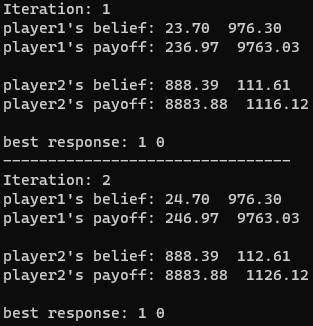
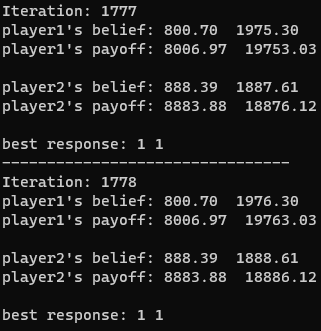
* + 1. Pure-Coordination Game

這個game有可能收斂到兩個pure-strategy NE (r1, c1)、(r2, c2)，也有可能在(r1, c2)、(r2, c1)之間一直跳，收斂到mixed-strategy NE，、、、。

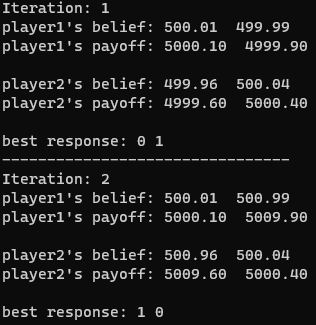
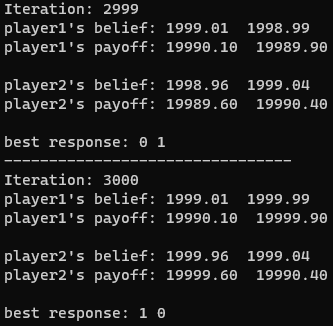
* + - 1. 收斂到pure-strategy NE (r1, c1)，自從game進入到(r1, c1)後，兩個players的payoff變動皆為(+10, 0)，這會使得接下來所有回合都選擇(r1, c1)。

* + - 1. 收斂到pure-strategy NE (r2, c2)，自從game進入到(r2, c2)後，兩個players的payoff變動皆為(0, +10)，這使得接下來所有回合都選擇(r2, c2)。

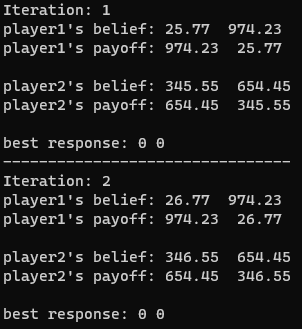
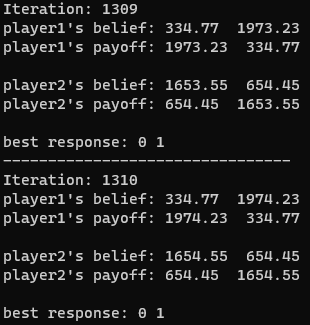
* + - 1. 收斂到mixed-strategy NE，如果起始belief接近、、、，但有些微不同，則會造成結果於(r1, c2)、(r2, c1)之間一直跳。因為各player之內的初始payoff差值小於10，在best response為(r1, c2)、(r2, c1)狀況下，都是造成比較小的payoff再加10，這樣上一輪payoff較大的一方在下一輪會變成比較小，如此不斷循環，最後結果為mixed-strategy NE。

* + 1. Anti-Coordination Game

這個game有可能收斂到兩個pure-strategy NE (r1, c2)、(r2, c1)，也有可能在(r1, c1)、(r2, c2)之間一直跳，收斂到mixed-strategy NE，、、、。

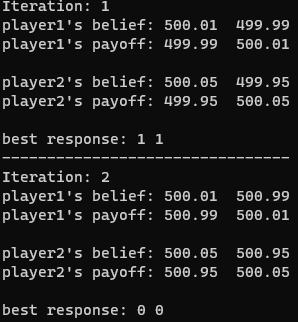
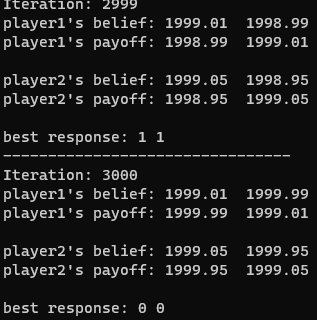
* + - 1. 收斂到pure-strategy NE (r1, c2)，自從game進入到(r1, c2)後，player1的payoff變動為(+1, 0)，player2的payoff變動為(0, +1)，這會使得接下來所有回合都選擇(r1, c2)。

* + - 1. 收斂到pure-strategy NE (r2, c1)，自從game進入到(r2, c1)後，player1的payoff變動為(0, +1)，player2的payoff變動為(+1, 0)，這使得接下來所有回合都選擇(r2, c1)。

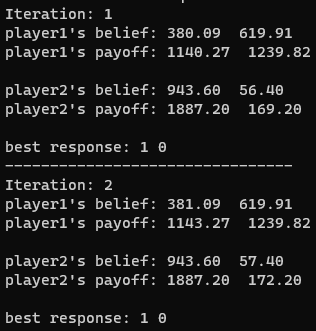
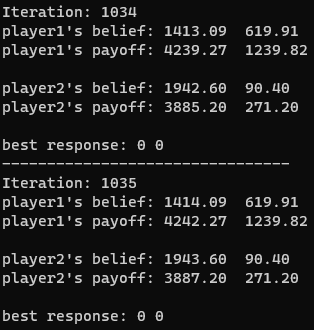
* + - 1. 收斂到mixed-strategy NE，如果起始belief接近、、、，但有些微不同，則會造成結果於(r1, c1)、(r2, c2)之間一直跳。因為各player之內的初始payoff差值小於1，在best response為(r1, c1)、(r2, c2)狀況下，都是造成比較小的payoff再加1，這樣上一輪payoff較大的一方在下一輪會變成比較小，如此不斷循環，最後結果為mixed-strategy NE。

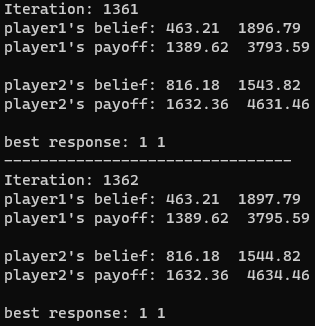
* + 1. Battle of the Sexes

這個game有可能收斂到兩個pure-strategy NE (r1, c1)、(r2, c2)，也有可能在(r1, c2)、(r2, c1)之間一直跳，收斂到mixed-strategy NE，、、、。

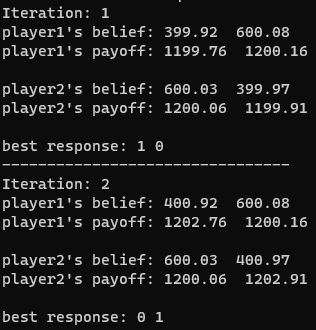
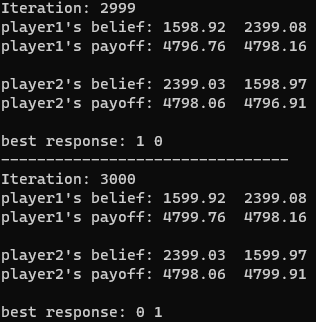
* + - 1. 收斂到pure-strategy NE (r1, c1)，自從game進入到(r1, c1)後，player1的payoff變動為(+3, 0)，player2的payoff變動為(+2, 0)，這會使得接下來所有回合都選擇(r1, c1)。

* + - 1. 收斂到pure-strategy NE (r2, c2)，自從game進入到(r2, c2)後，player1的payoff變動為(0, +2)，player2的payoff變動為(0, +3)，這使得接下來所有回合都選擇(r2, c2)。

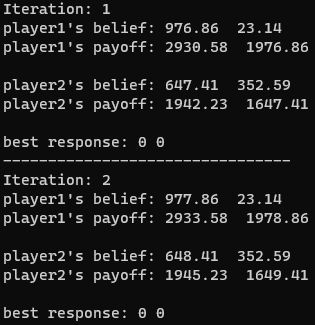
* + - 1. 收斂到mixed-strategy NE，如果起始belief接近、、、，但有些微不同，則會造成結果於(r1, c2)、(r2, c1)之間一直跳。因為各player之內的初始payoff差值小於2，在best response為(r1, c2)、(r2, c1)狀況下，都是造成比較小的payoff再加2或3，這樣上一輪payoff較大的一方在下一輪會變成比較小，如此不斷循環，最後結果為mixed-strategy NE。

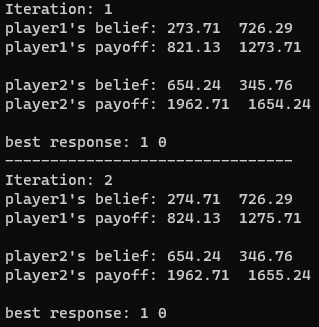
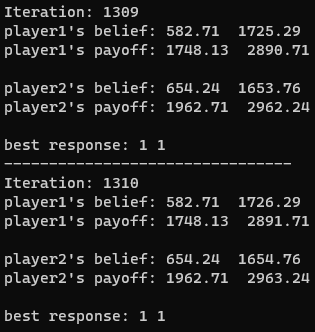
* + 1. Stag Hunt Game

這個game有可能收斂到兩個pure-strategy NE (r1, c1)、(r2, c2)，也有可能在(r1, c2)、(r2, c1)之間一直跳，收斂到mixed-strategy NE，、、、。

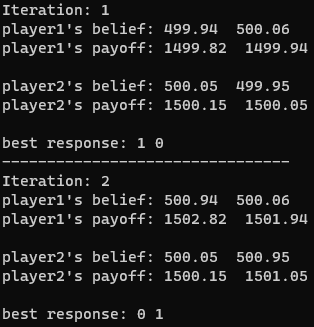
* + - 1. 收斂到pure-strategy NE (r1, c1)，自從game進入到(r1, c1)後，兩個players的payoff變動皆為(+3, +2)，這會使得接下來所有回合都選擇(r1, c1)。

* + - 1. 收斂到pure-strategy NE (r2, c2)，自從game進入到(r2, c2)後，兩個players的payoff變動皆為(0, +1)，這使得接下來所有回合都選擇(r2, c2)。

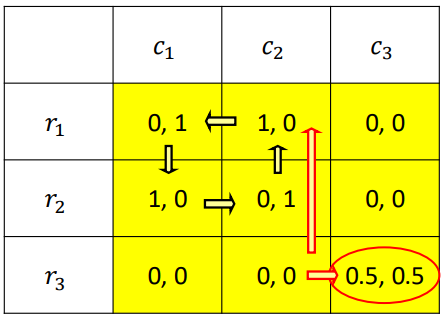
 

* + - 1. 收斂到mixed-strategy NE，如果起始belief接近、、、，但有些微不同，則會造成結果於(r1, c2)、(r2, c1)之間一直跳。因為各player之內的初始payoff差值小於1，在best response為(r1, c2)時，player1 payoff 變動為(0, +1)，player2 payoff 變動為(+3, +2)；在best response為(r2, c1)時，player1 payoff 變動為(+3, +2)，player2 payoff 變動為(0, +1)。可以觀察出所有cases都是較小的payoff比較大payoff再多加1，這樣上一輪payoff較大的一方在下一輪會變成比較小，如此不斷循環，最後結果為mixed-strategy NE。

* + 1. Observation and Conclusion

Fictious play不是永遠都可以找到NE，假設utility matrix如下圖。



pure-strategy NE為(r3, c3)，mixed-strategy NE為、、、、、。

一旦fictious play進入了(r1, c1)、(r2, c1)、(r2, c2)、(r1, c2)任何一個state，他就會一直無限的被困在這個infinite-loop中，沒有辦法到達pure-strategy NE(r3, c3)，同時他也不會收斂到、、、、、。