人工智慧 作業三

資工112 40847015S 紀軒宇

1. 請說明你所使用之機器軟硬體規格、所用的作業系統、開發之程式語言版本

等相關資訊以及你為何選擇這樣的規格。另外請提供你的連絡電話，以便不

時之需。

CPU: Intel® Core™ i7-8750H

GPU:Nvidia GeForce GTX 1060

ram:24g ddr4 2666

OS: Windows 10

開發軟體：Python 3.9.6

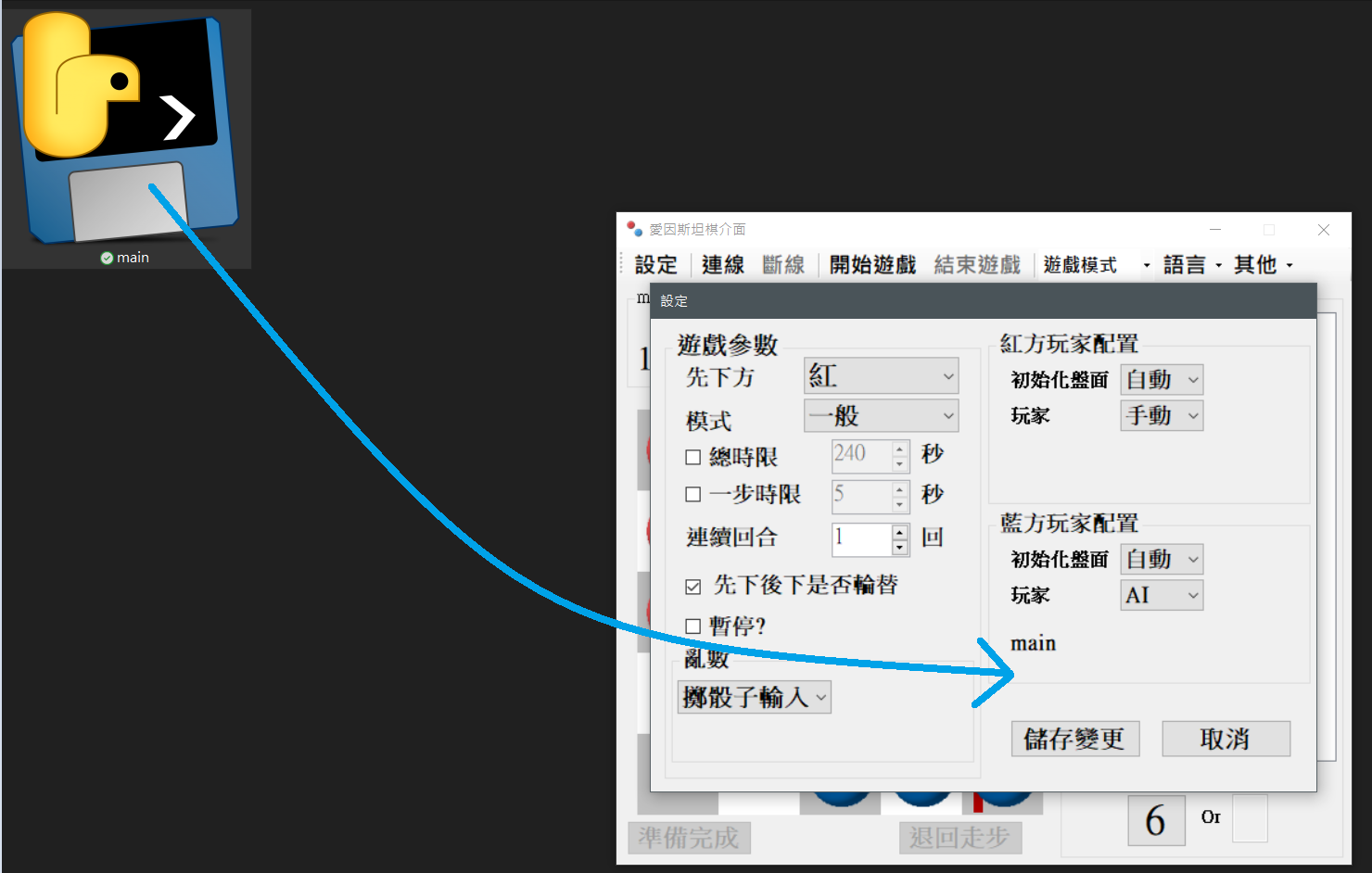
會選擇這樣的規格是因為當時考量未來可能會學習機器學習相關知識，選擇了有獨立顯示卡的電腦，方便日後學習。

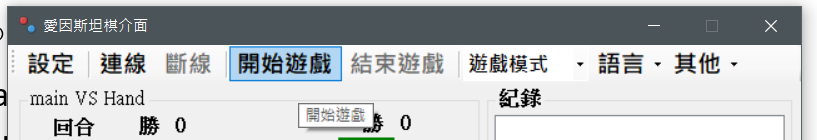
開發軟體：Python 3.9.6

連絡電話：0988-056-412

2. 如何執行程式

這次的程式結合教授研究生林品儒愛因斯坦棋平台，使用方法，先將資料夾中的ECUI打開，點擊左上角之設定，並將main.exe拖曳至一方的AI玩家，接著左方可選擇先下後下、亂數種類可選擇、也可以切換模式，觀察程式如何運作。

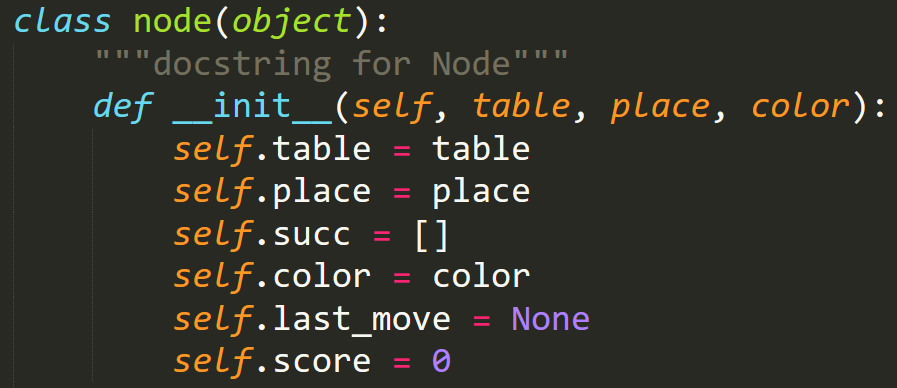


接著回到主頁面，按下開始遊戲即可開始遊戲。

　　其餘詳細操作方法可見由研究生林品儒撰寫的介面使用說明

3. 你使用甚麼方法、甚麼資料結構、甚麼技術(操練要項)，來解決這個問題，並請你說明你測試一些盤面時的表現如何。

- 棋盤表示方法



這邊使用

table為一個1\*25的陣列，分別標註個棋子編號，代表棋盤

place為一個1\*12的陣列，分別表示個顏色棋子的位置

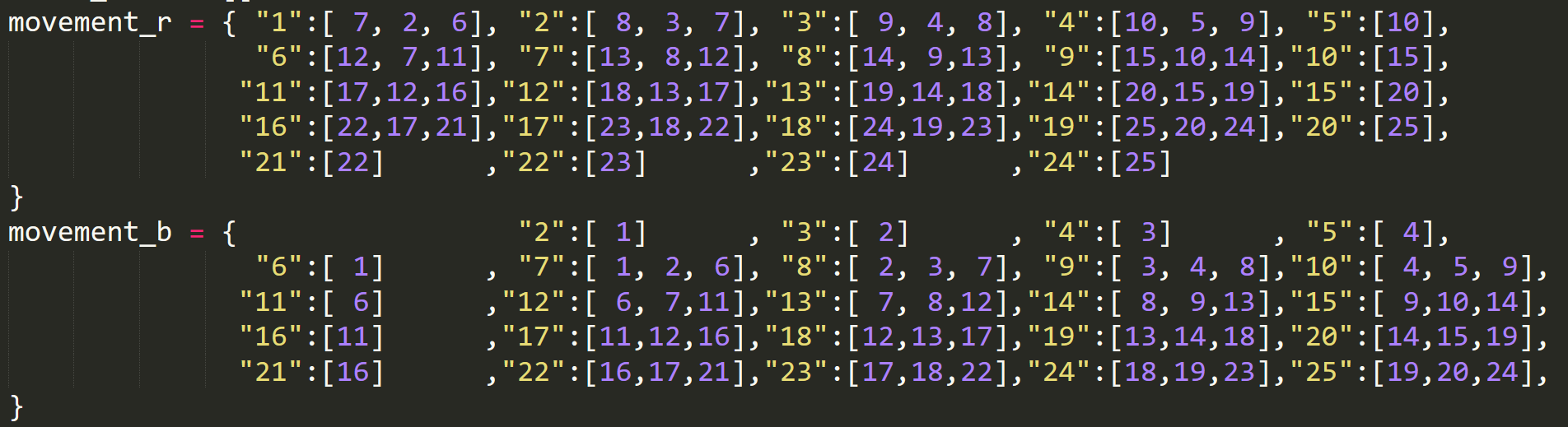
succ為一個不定大小的陣列，裡面存放該盤面的各種可能

color為該盤面下一個要下棋的顏色

last\_move為一個1\*2的陣列，代表移動的前後點的座標

score為該盤面的審局函數之分數

- 走步產生

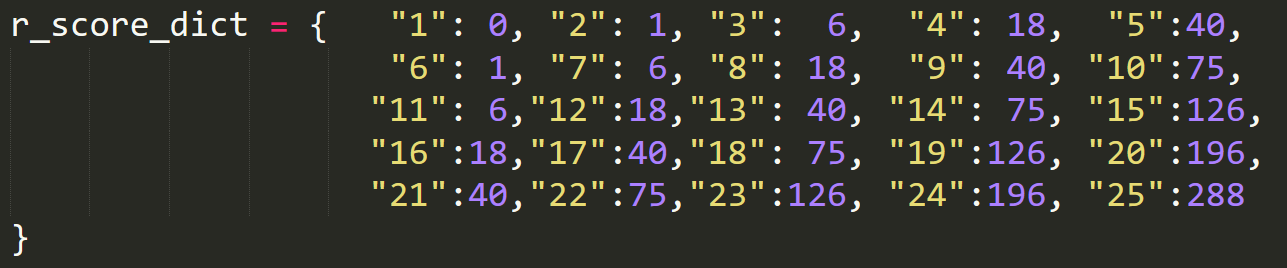


因為愛因斯坦棋只有25個位置，所以直接使用dict的資料結構，相對檢查邊界等複查的方法，此資料結構因為是使用hash table查找，所以速度會快上非常多。

決定了可以走哪些步之後，接下來就是產生新的盤面，新盤面的部分，直接將棋盤上指定座標的點換成新的點後，原座標的點更換成0即可，接著回去更新place的內容。

- 審局函數

審局函數部分，每個顏色個別隊每個點給其一個分數並計算其平均，如紅方的分數對應如下，這邊採用dict也是計算速度考量，其餘部分如果剩餘棋子>4顆，每有一個棋子就-100分，最後回傳兩顏色間分數的差，如果有能讓棋局結束的狀況（如將對方棋子全部吃完或走到終點），就直接+10000分。



- 尋找走步

這邊先建立一棵樹，在之後用來做Alpha-Beta Pruning，接者使用Alpha-Beta Pruning，並在traceback時更新各節點最大（最小）的score。

- 盤面表現

(i) 如此初始盤面，紅方（AI）下，擲到4，此時因為吃掉自己的棋子可以增加其他棋子可以選擇的機率，所以會先吃掉自己原先在位置7個5號棋子。



(ii) 再來是這個盤面，紅方（AI）下，擲到3，在沒有其它吃子情況下，就會往右下方移動。



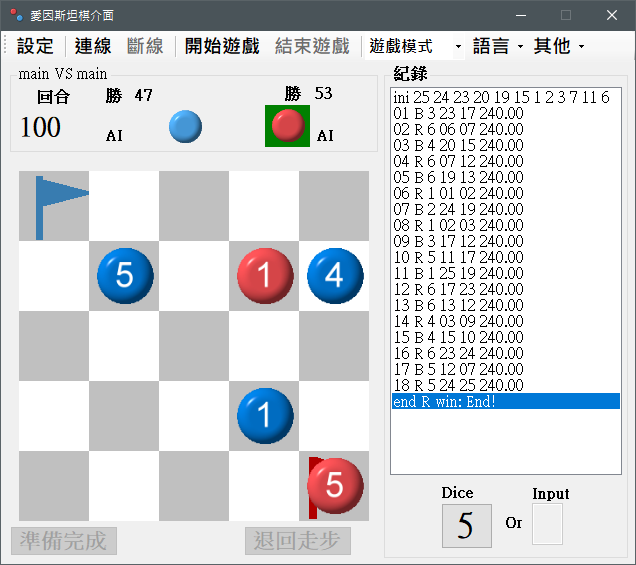
(iii) 這個盤面，藍方（AI）下，擲到6，雖然往左上角走有機會會被紅方的3號吃掉，但是因為還有5顆棋子，對於藍方被吃掉反而可以增加整體的靈活性，所以直接往左上角走。



(iv) 進階一點的狀況，藍方（AI），擲到5，為了避免被紅方的1吃掉，選擇吃掉自己的3號來前進



(v) 最後，讓同個AI分別作為紅藍方輪流對下100局，可以看出勝率大約都接近50%，表示可以通用於紅方及藍方。



4. 請說明你做此作業所踫到的一些狀況及困難。

這次是第一次以別人開發好的介面做進一步的開發，最大的困難是要先理解當初開發人設計這些輸入輸出規定的用意，將那些輸入做為自己的輸入資料，再將輸出依照介面的規定輸出，照這樣開發的最大好處，就是因為是已經做好的介面，通常都會避免掉一些錯誤輸入，在開發時就比較少需要去注意錯誤輸入的問題。

5. 請列出你的參考文獻(含網站)來源，並請說明參考了那些部份用於作業中。

- Alpha-Beta Pruning實作部分參考：https://www.javatpoint.com/ai-alpha-beta-pruning