hw3

學生 : 40947012S 黃至瑜

1. 說明
   - OS : Windows
   - Language : Python，因為操作容易，而且支援套件多，可直接用pygame做簡易遊戲
   - 電話 : 0935001891
2. 基本操作版code為**chess.py**，註解在程式碼內。

   加分介面版code為**GUI.py**，註解在程式碼內。

   直接在終端機執行python
3. code為**random_input.py**

   n, m在長度1到8之間，隨機產生0與1填滿矩陣，若遇到全為0的矩陣重複產生直到出現1為止。

   直接寫進**input.txt**，此程式只產生隨機input。

   建議測資 :

```
# 第一個測資
1 2
1 0
# 第二個測資
3 4
1 0 0 1
0 0 0 1
1 1 1 1
# 第三個測資
3 4
1 1 1 1
0 0 0 0
1 1 1 1
# 第四個測資
4 8
1 1 1 1 0 0 0 0
0 0 0 1 1 1 1 1
0 0 0 1 1 1 1 1
1 1 1 1 0 0 0 0
# 第五個測資
7 5
0 1 1 1 1
0 0 0 0 1
0 1 1 1 0
0 0 0 1 0
0 1 0 1 1
1 0 0 0 1
0 1 1 0 1
# 第六個測資
6 8
0 0 1 0 0 0 0 1
```

```
0 0 1 0 0 0 0 1
0 0 1 0 0 0 0 1
0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 1
```

4. 基本操作版code為**chess.py**

   方法與資料結構 : 建立class Node來存取各節點資訊，包含盤面、選擇行列及位置、分數、父節點、子節點、路徑等資訊。用get_successors()找出所有子節點，以分數由大到小排序，再以Node形式存起來。

   操練要項 : 盤面都是使用0和1表示，1表示棋子位置。走步與產生child都是處理在get_successors中，先選擇各行與各列，再產生可能的結果盤面。在alpha beta pruning中，每當找不到子節點，計算分數並回傳。其餘則用遞迴搜尋，進行alpha剪枝或beta剪枝。結果包含選擇第一步、最後分數與時間輸出在**output.txt**

   盤面表現與結果 (例子為以上建議測資)

input	score	first step	time
1 2 1 0	1	Row #: 1	0.000
3 4 1 0 0 1 0 0 0 1 1 1 1 1	3	Row #: 3	0.007
3 4 1 1 1 1 0 0 0 0 1 1 1 1	0	Row #: 1	0.000
4 8 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0	2	Column #: 4	0.275
7 5 0 1 1 1 1 0 0 0 0 1 0 1 1 1 0 0 0 0 1 0 0 1 0 1 1 1 0 0 0 1 0 1 1 0 1	3	Column #: 5	0.281
6 8 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1	3	Column #: 8	0.014

> 💡 在盤面棋子較多且較複雜時，所需時間較多

功能和優點 : 用class Node記錄每個節點的完整資訊，tree的各節點都可隨意搜尋，因此想要查看某節點的任意資訊都很方便。另外，在加分介面版中，有呈現先手及後手選擇的一種完整路徑，並且顯示累計分數，最後結果直接呈現在介面上。使用者使用非常直觀，只須點選Game Start按鈕。

5.

- https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-4-alpha-beta-pruning/
    - alpha beta pruning的pseudocode
- https://blog.csdn.net/qaqwqaqwq/article/details/127476404
    - alpha beta pruning的Python code
- https://www.uj5u.com/qita/293051.html
    - pygame製作遊戲介面

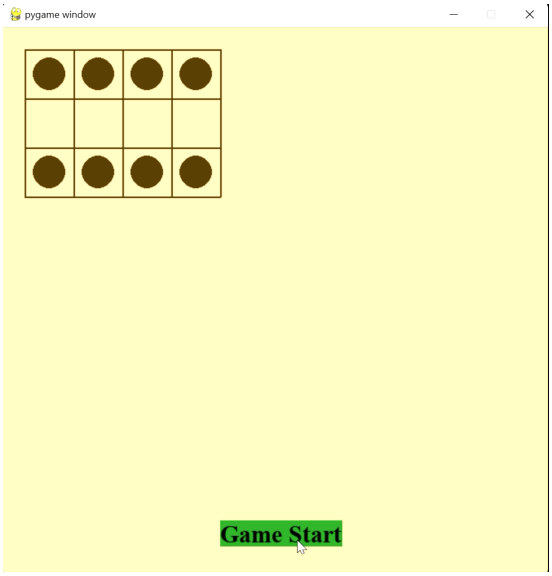
6.

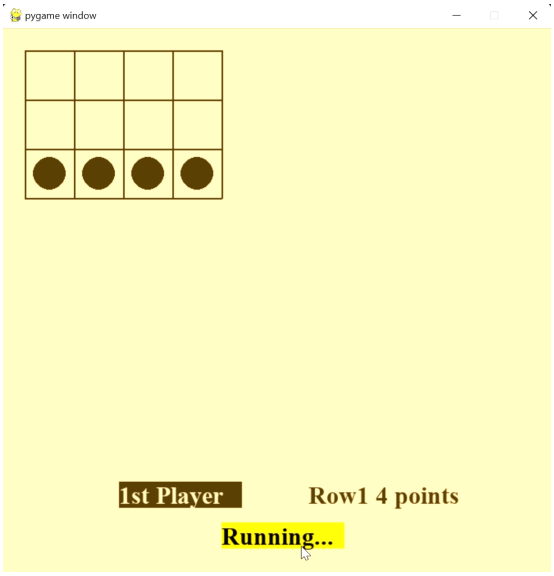
- 狀況1 : 原本在實作alpha beta pruning，不知道是要用後面幾步的預測結果作為回傳，所以用第一步可以選擇的最大棋子數作為結果。後來發現整個想法錯誤，因而耽誤一些時間。
- 狀況2 : 曾經嘗試將完整的tree建立出來，後來發現會占用許多不必要的空間以及增加額外時間，由於alpha beta pruning會剪掉不必要的分枝，所以在開始建立完整tree是不必要的。
- 狀況3 : 雖然找到最佳解，但無法找到選擇的完整路徑。方法是在Node加上path紀錄此節點走過的路。
- 狀況4 : 在少部分案例中，雖然分數結果是正確的，但是路徑並非最佳解，在get_successors中增加分數由大到小排序後，即可改善並有效減少時間。

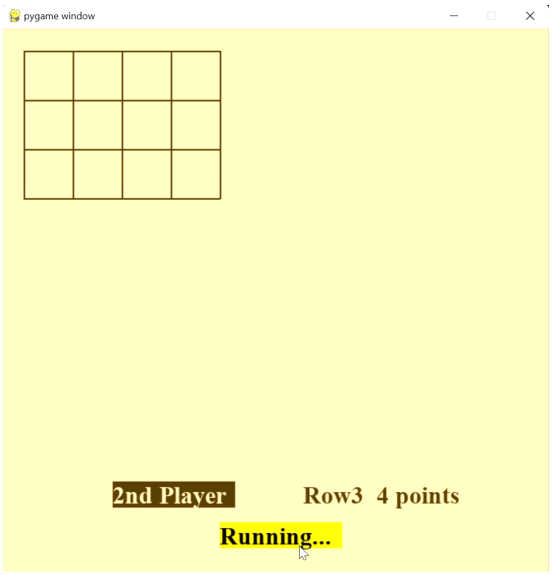
7. 加分介面版code為**GUI.py**

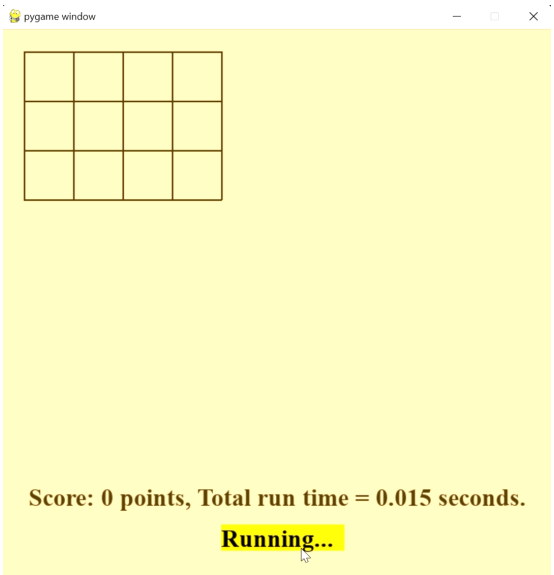
   使用pygame製作的棋子遊戲介面版，可以實現兩玩家選取棋子過程 (選取位置和玩家的累計分數)、分數、時間，因為介面設定每1.5秒才會走一步，避免無法有效地呈現選取過程，所以此程式的時間輸出是紀錄運作alpha beta pruning不包含後來設定的時間限制。此程式讀取**input.txt**，結果直接呈現在介面，也會輸出至**output.txt**，因為前面提及的時間計算問題。若要查看精確輸出結果可以用**chess.py**。

   以下為介面截圖 (有部分色差) :


pygame window
Game Start


pygame window
1st Player
Row1 4 points
Running...


pygame window
2nd Player
Row3 4 points
Running...


pygame window
Score: 0 points, Total run time = 0.015 seconds.
Running...