AI Chess

新竹高中 208班17號 陳宇彥

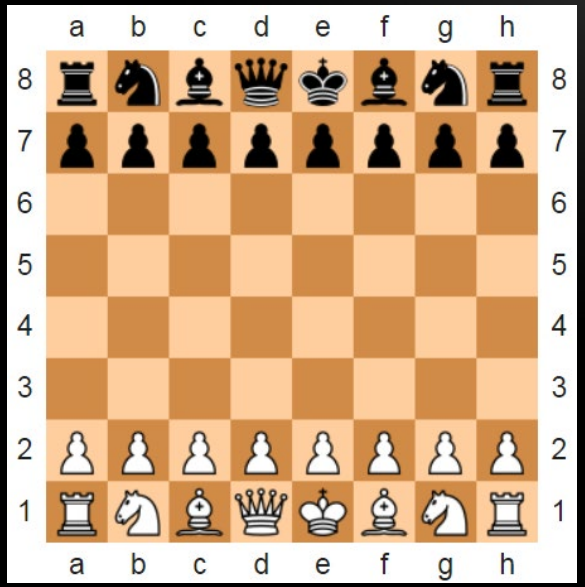
Outline

- Chess Board & Pieces Moves
- Specific Rules & Competition Rules
- AI Algorithm
- Conclusion

Chess Board & Pieces Moves

- Set Up the Board
- Pieces
- The King
- The Queen
- The Rook
- The Bishop
- The Knight
- The Pawn

Set Up the Board



棋子的擺法

棋局開始前，應將棋子擺在初始位置上。
棋子的初始位置如左圖。
其中要注意：

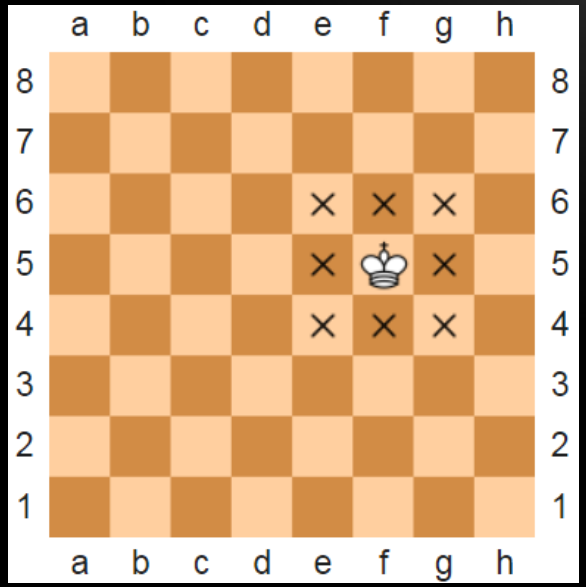
1. 白棋應擺在棋盤的第1、2行，黑棋應擺在棋盤的第7、8行。
2. 王所在的格的顏色與王自身的顏色不同。

Pieces

Piece	King	Queen	Rook	Bishop	Knight	Pawn
Number	1	1	2	2	2	8
Symbols	♔ ♚	♕ ♛	♖ ♜	♗ ♝	♘ ♞	♙ ♟

國際象棋的棋子分為兩種顏色：淺色棋子稱為「白棋」；深色棋子稱為「黑棋」。對弈雙方各有16枚棋子，分別為一王(King)、一后(Queen)、雙象（或主教）(Bishop)、雙馬（或騎士）(Knight)、雙車（或城堡）(Rook) 和八兵(Pawn)。

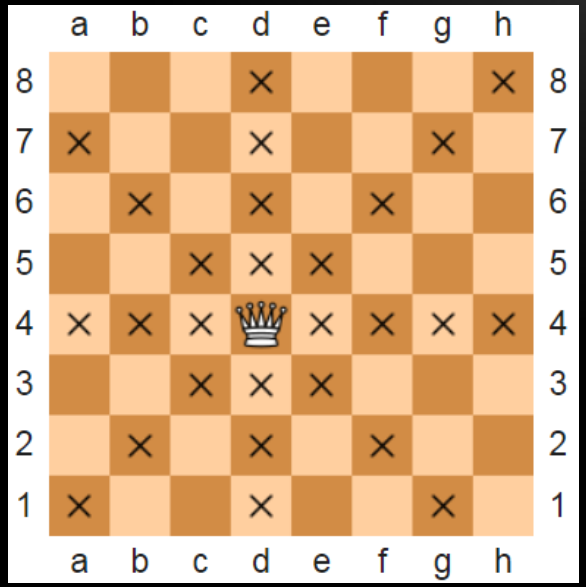
The King



國王（King）

又稱為「王」，是整個棋局中最為重要的棋子，不能被吃。橫、直、斜走均可，但每次只能走一格，所走到的位置不可有對方棋子的威脅，吃子與走法相同。

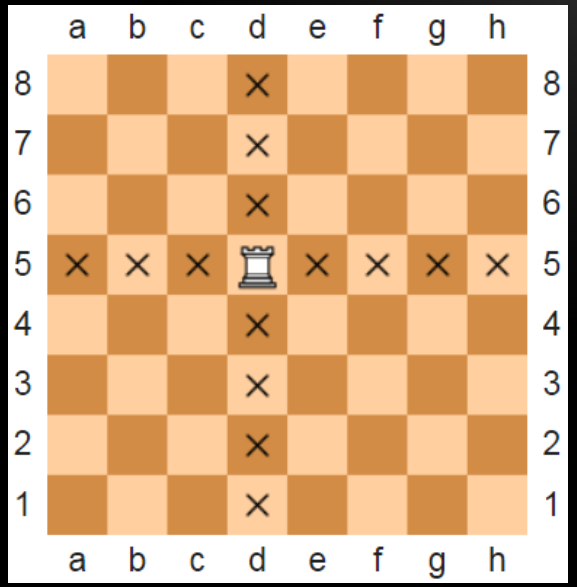
The Queen



皇后（Queen）

又稱為「后」，是威力最強的棋子。橫、直、斜走均可，格數不限，但不可越過其他棋子。吃子和走法相同。

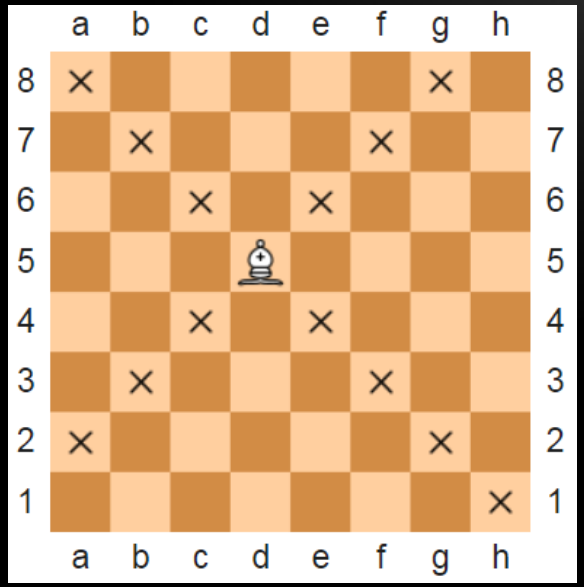
The Rook



城堡（Rook）

又稱「車」，一如中國象棋，走法是橫走或直走，格數不限，但不可斜走，也不可越過其他棋子。吃子與走法相同，另外有一種特殊的走法，稱作「王車易位」(castling)。

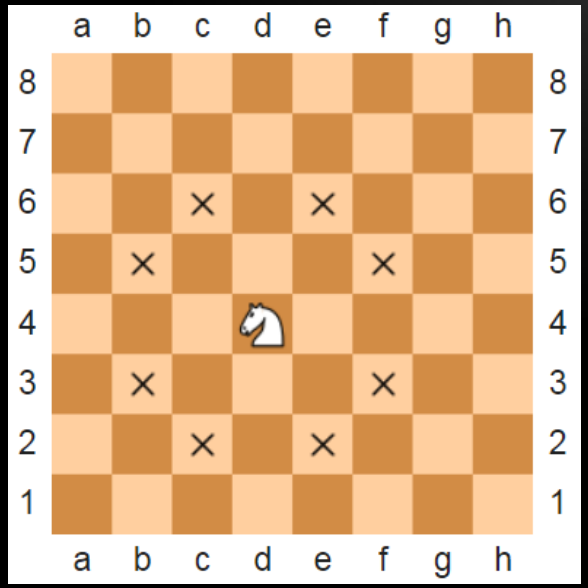
The Bishop



主教（Bishop）

又稱「象」，只可斜走，格數不限，但不可轉向，也不可越過其他棋子。吃子與走法相同 。開局時雙方各有兩主教 ，一在白格，一在黑格，白格主教只能在白格內移動，黑格主教只能在黑格內移動。

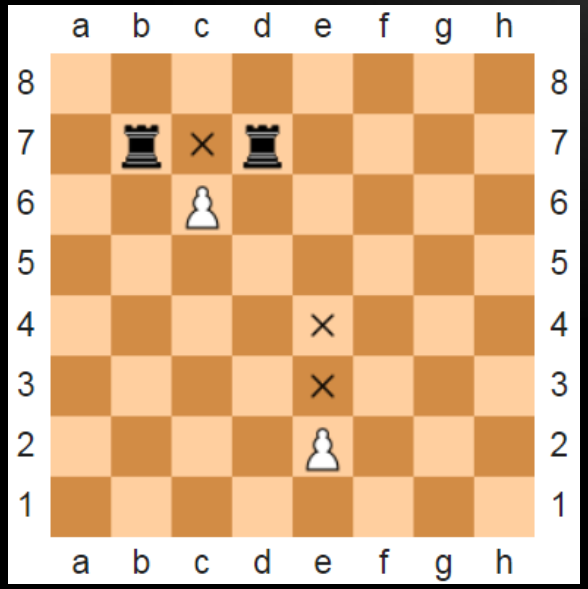
The Knight



騎士（Knight）

又稱「馬」，走法和中國象棋的馬相同，同樣是走「日」字，或英文字母大寫的「L」形：即先向左（或右）走1格，再向上（或下）走2格；或先向左（或右）走2格，再向上（或下）走1格。西洋棋的騎士沒有「絆馬腳」的限制，故騎士可越過其他棋子。吃子與走法相同。

The Pawn



兵（Pawn）

士兵第一步可前進一格或兩格，以後每次只能前進一格，不可向後退或越過其他棋子。但吃對方棋子時，則是向位於斜前方的那格去吃，另有一種特殊走法「吃過路兵」(en passant)。

Specific Rules & Competition Rules

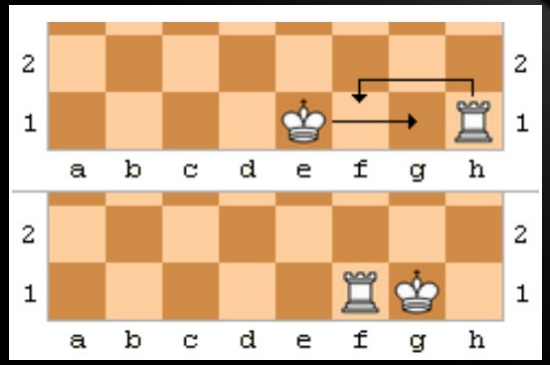
- Pawn Promotion
- Castling
- En Passant
- End of the Game
- Recording Moves

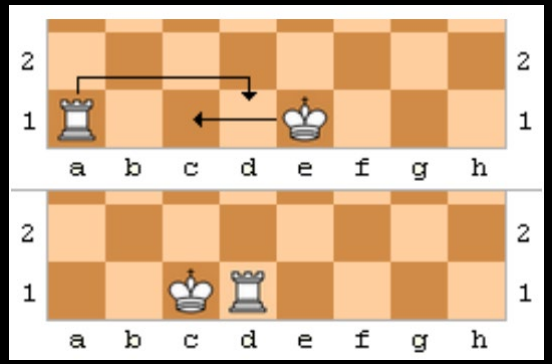
Pawn Promotion

兵的升變

一方的兵從起始位置移動到對方的最後一行（底線）後，可升變（promote）為己方除王和兵之外的任何棋子。升變的棋子種類由棋手自由選擇，不受棋盤上現有棋子的限制。

Castling





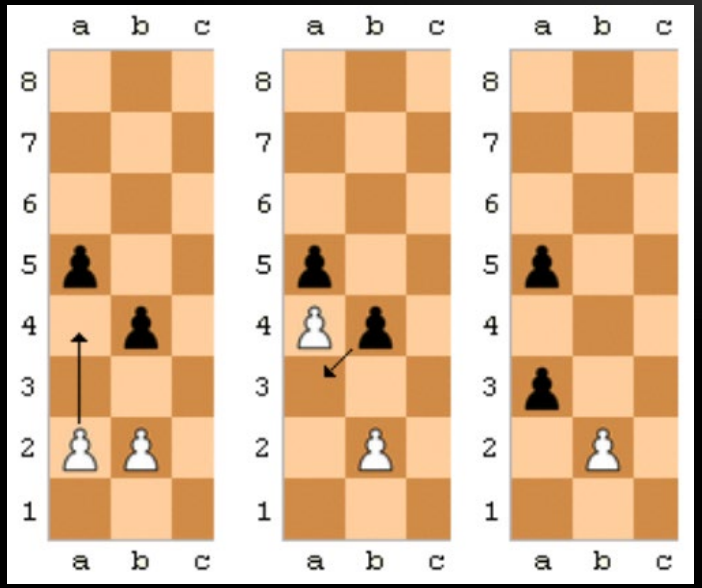
王車易位

王車易位（castling）是一方王和車的位置調整。調整王翼的車叫王翼易位（短易位），調整后翼的車叫后翼易位（長易位）。走法是將王向參與易位的車方向移動兩格，再將車越過王，放在緊鄰王的一格。

有以下情形之一時，王車不得易位：

1. 王或參與易位的車以前曾移動過（無論是否回到了原來的位置）。
2. 王正在被將軍。
3. 易位時王要經過的格正受到對方的進攻。
4. 易位後王將佔據的格正受到對方的進攻。
5. 王與參與易位的車之間有其他棋子阻隔。

En Passant



吃過路兵

當一方的兵從原始位置向前一步走兩格時，如果所到格同一橫線的相鄰格有對方的兵，則後者可以立即吃掉相鄰的前者，但是占據原來位置的斜前方那一格，而不是前者原來占據的那格（換言之，後者此時可認為前者只行進了一格，從而直接斜進將其吃掉）。「吃過路兵」必須立即進行，否則即喪失權利。

End of the Game

有以下情形之一時，判該方作負：

1. 一方的王被將死 (checkmate)。
2. 比賽中，一方用完時間。
3. 一方認輸（通常是因為該方的王走投無路，或該方已看出自己敗勢已定）。
4. 一方退出遊戲，等同認輸。

有以下情形之一時，判雙方和棋：

1. 雙方同意和局。提議作和者要等對方完成其步，和議被接受後不得反悔。
2. 雙方均沒有足夠的棋子在有限步數內將死對方。
3. 比賽中，雙方均用完時間。
4. 一方用完時間，但對方在有限步數內沒有足夠的棋子將死該方。
5. 行棋的一方在未被將軍的情況下無子可動（稱為「逼和（stalemate）」）
6. 在連續的50個回合內，雙方沒有吃子也沒有移動過兵。
7. 下一步之前，同樣的局面出現或即將出現三次或以上。

Recording Moves

現今西洋棋通用的記譜方式為代數記譜法，基本規則為先記錄移動的棋子名，若移動的棋子是兵時通常省略不記；接著再寫其所到格之座標，若有吃子通常會前加上 x 。如果兩個相同的棋子都可能移動到目標格子，則依情況寫出來源格的行或列避免岐義。此外 0-0 表示短易位，0-0-0 表示長易位，兵升變時以 = 表示升變的棋子。

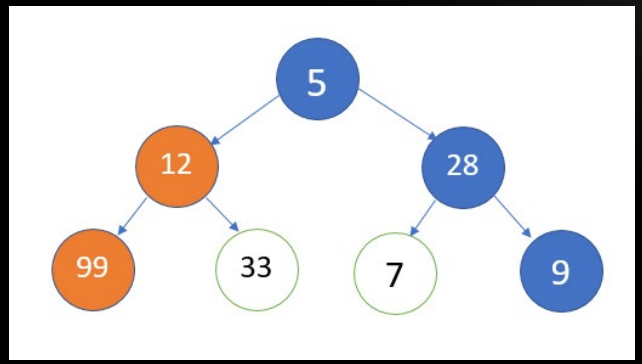
其他一些常用的記錄棋譜的記號還有：

e.p.：表示吃過路兵。
+：表示將軍。
++：表示雙將。
#：表示將死。
1/2-1/2：表示棋賽的結果是和棋。
1-0：表示棋賽的結果是白方勝。
0-1：表示棋賽的結果是黑方勝。

AI Algorithm

- Greedy Algorithm
- MiniMax Algorithm
- NegaMax Algorithm
- MiniMax with Alpha-beta Pruning

Greedy Algorithm

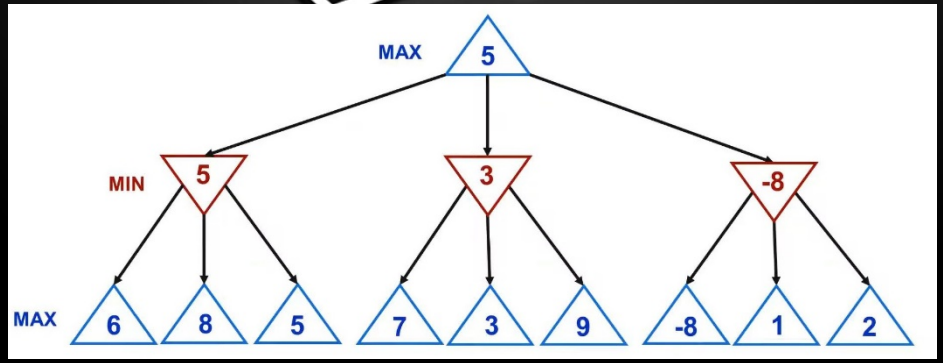


貪婪演算法（greedy algorithm），是一種在每一步選擇中都採取在當前狀態下最好或最佳（即最有利）的選擇，從而希望導致結果是最好或最佳的演算法。

做法：對每個棋子給與一個分數，愈重要的棋子分數愈高。將白方分數以正值計算，黑方分數以負值計算，AI會根據每個可能的下一步都算出其總分，以分數絕對值最高的那一步來決定下一步的選擇。

從左圖可知，greedy algorithm不見得會得到最佳的結果。

MiniMax Algorithm



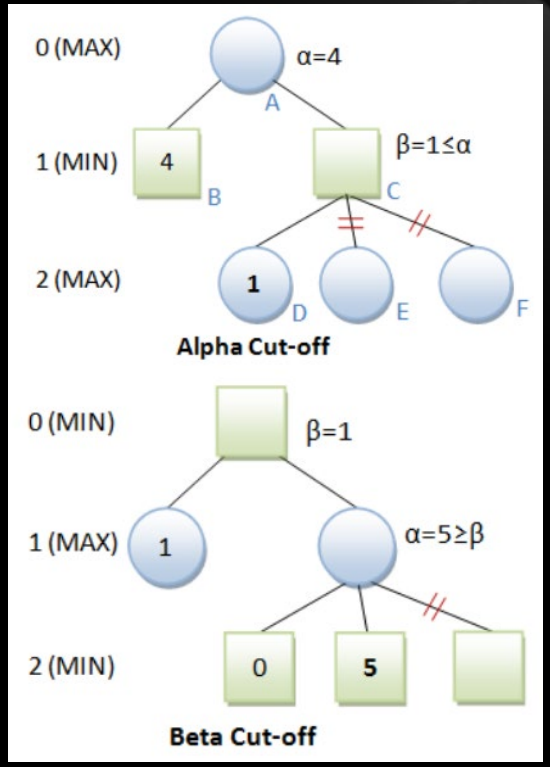
MiniMax演算法使用先前Greedy演算法相同的評分系統，只不過現在我們要增加評估對手的反應來增加贏的機率。此時我們要在賽局樹上增加分數計算的階層；以MAX賽者追求極大化分數，MIN賽者追求極小化分數的原則，來找到最佳的下一步。

根據上圖， MIN賽者對於其下一階層所評估的分數分別為5, 3, -8，所以其上一層的MAX賽者應該根據MIN賽者所評估的分數來選擇最左邊的5。

NegaMax Algorithm

在做MiniMax的搜尋時，會有必須交替選擇最大及最小值的困難，因此加上負號，使得在搜尋對方的評估分數時也以最大值來判斷，則搜尋賽局樹就只需搜尋最大值即可，也可以方便程式上的編寫。

MiniMax with Alpha-beta Pruning



在做MiniMax的搜尋時，在MAX賽者階層節點的Alpha值已經大於其子節點的Beta值時，就不可能選擇這個子節點來著手，所以就可以不用展開此節點，這個技巧稱為Alpha切割；相對的，當MIN賽者階層節點的Beta值已經小於其子節點的Alpha值時，就可以不用展開此節點，稱為Beta切割。對於MiniMax搜尋而言，子節點的展開順序並不重要，但對Alpha-beta Pruning來說，越能先展開較好的子節點，就越能刪除更多的節點，對於節省搜尋時間有顯著的幫助。

Conclusion

這個自主學習計畫，可說是從我上了高中就已經展開了。那年暑假我去學習了python這個程式語言，對於程式設計有了初步的認識。從計畫開始之後，我開始仔細地去了解西洋棋的規則，以及製作西洋棋程式該有的知識基礎。經過不斷地努力嘗試，終於完成了所有下棋的基本規則，與特殊規則；其中包括了Pawn Promotion、Castling、En passant，與終局checkmate與stalemate的判斷，另外還加上了能夠記錄整個棋局過程的記譜功能，實現了西洋棋的基本功能。

這個計畫的另一個重點是AI，我從網路上學習了如何讓AI自動作出下一步的判斷，從最基礎的Greedy Algorithm開始，學習各種Algorithm的改善演進，經過MiniMax、NegaMax，再到MiniMax with Alpha-beta pruning，逐步確認AI判斷結果與執行時間的改善。在整個計畫過程當中，我從網路資料學習到了許多新的知識，也讓我感受到，只要努力，就可以找到問題的答案。