第七組 期末專案書面報告

B084020016陳育綺

B084020023朱曼華

B084020032王郁文

B084020034謝旻臻

B084020043蔡昀芳

B084020048許映婷

B084020051丁海倫

**壹、研讀資料**

一、AI部份

我們根據網路上流傳的「權重表」為我們設計AI的主要依據之一，根據各個點所對應的數值大小不同，讓AI做出最有利的選擇。數字越大，代表此點最為有利，通常是在四個角落會出現最大值，在四個角落的周圍會出現最小值，因為在這些地方落子，會讓對手有極大的機會落子在四個角落，此為我們的參考網站《課程設計 --- 黑白棋中的 AI》<https://www.itread01.com/articles/1475964375.html>。

二、美化資料

1. 利用#include<graphics.h>中的circle(x, y, radius)函式，將棋子變成圓形，其中(x, y)代表這個圓的圓心，radius則代表這個圓的半徑，此為我們的參考網站《Draw circle in C graphics》https://www.geeksforgeeks.org/draw-circle-c-graphics/。

2. 利用#include<stdio.h>中的setcolor()函式，將背景、棋子改成不同的顏色，此為我們的參考網站《setcolor function in C》https://www.geeksforgeeks.org/setcolor-function-c/。

3. 使用entername()讓雙方可以輸入名字，並在遊戲進行石取代玩家1/玩家2，讓玩家可以更加清楚得知遊戲訊息。

4. 在遊戲中加入音樂，我們覺得在遊戲中加入音樂，可以更帶動氣氛，讓玩家玩得更有興致，此為我們參考的網站《How to insert (background) music with c language》<https://www.programmersought.com/article/44423247819/>。

**貳、程式介紹**

一、PPT內容說明

1. 功能介紹 : 共分為三種，分別是玩家對玩家、玩家對AI，及AI對AI。後兩者對AI的模式都可以選擇「困難」或「簡單」。

2. 遊戲介面 :

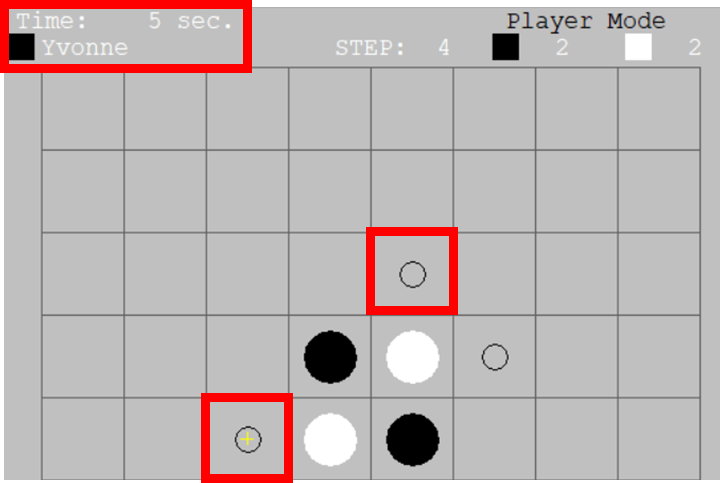
(1) 以黑色圓型表示黑棋，白色圓形表示白棋。

(2) 黑色空心圓圈表示可下棋的位置。

(3) 黃色十字代表目前位置。

(4) 左上角顯示目前的輪次為的玩家名字及開局秒數。

(5) 右上角顯示當前的模式及檯面上雙方的棋子數。



3. AI 介紹

(1) Easy mode

簡單模式使用if判斷式以及計算可翻棋子數的函式reverse()，當下在該點可以翻出台面上最多棋子時，AI自動斷定該點為最佳落點。實作上，先將符合規則的合法落點儲存成鏈結串列，指標指向串列的第0個值，依序讀進reverse()中，當可翻棋子數大於暫存最大值，即更新最大值，並將指標往後移動一格，直至指標指向NULL。鏈結串列中儲存的為指標的x, y座標及連向下一個的指標，在程式一開始使用typedef struct()自行宣告名為loc的資料型態。

(2) Hard mode

保留簡單模式中計算最大可翻棋子數的部分，另外根據資料中提及，依許多玩家的經驗法則所給出的權重表建構出一個二維陣列expectnum[][]，儲存每個位置對應的權重分數。分數越高即代表下在該位置越有利。而在判斷時，綜合權重表的分數以及最大可翻棋子數，判斷下在哪邊更為有利。此模式以權重分數為主，當權重相同時再考慮最大可翻棋子數。

二、輸入名字

1. 與玩家名字相關之宣告:

(1) function entername(): 供玩家輸入名字的函式。

(2) char msg3[40]: 區域變數，易更動。用於entername()，預先存入「Enter Player1 name:」共19字元，剩餘21字元用於存取玩家一的名字。

(3) char msg4[40]: 區域變數，易更動。同msg3，存取玩家二名字。

(4) char player1name[50]: 全域變數。已輸入的玩家一名字。

(5) char player2name[50]: 全域變數。已輸入的玩家二名字。

(6) int namecount: 使用於entername()，作為當前輸入字元於陣列msg3或msg4中的位置的指標。

(7) int name1count: 全域變數。作為陣列player1name的位置指標。

(8) int name2count: 全域變數。作為陣列player2name的位置指標。

2. entername()程式概述

使用用於存取操作按鍵的char key來記錄使用者按下的字元，以下考慮三個可能:

(1) 空白字元: 離開輸入名字的介面。

(2) 英文大小寫或數字: 將先前已經輸入的玩家名連同「Enter Player1 name:」等字元，利用函式庫strcat()合併入msg3陣列，再存入最新輸入字元進msg3和play1name。

(3) 退格鍵\b : 利用name1count之全域變數，找到目前輸入的位置，刪除先前一字元並令name1count指的位置後退一格。

結束玩家一名字輸入，先設一條件確定使用者目前為玩家對戰玩家的模式，再使用同一方式供玩家二輸入名字。

3. 初始化

當使用者於遊戲中按下「重啟遊戲」或「離開遊戲」按鈕，將會刪除原player1name和player2name內容，供第二次遊戲使用。

**參、檢討與改進**

一、動態改變權重表

我們使用的是在網路上找到的權重表來作為AI要下哪個位置的依據，但是那張權重表的問題在於，它是在場上無任何棋子下所劃分出來的分數，僅僅憑藉著這個位置的好壞而得出來的估值，並沒有考慮到當場上棋局發生變化時，每個位置的值也應該要有所改變，也許在無子情況下原本為較佳落子的位置，因為大局的變化而失去了優勢地位。

所以我們認為接下來的改善方式應該要把固定值的權重表改為動態式的，隨著棋局的改變，使用不同估值的權重表，交由AI判斷什麼類型的棋面，應該使用什麼樣子的估值表，或是隨著玩家落子的位置，將附近位置的AI可落子點做估值上的增減，再配合我們原本就有寫入的「選擇可翻最多棋子為落子點」的策略，相信可以做出比目前還要更聰明的AI。

二、Minimax + Alpha-Beta剪枝法

除了權重表外，其實還有其他的演算法是我們這組聽完其他同學demo後，認為可以加入AI取代權重表的方式。例如：Minimax演算法。這是一個在下棋遊戲中非常常見的演算法，透過選擇對自己最有利而對對手優勢最小的步路，以此來盡量達到對自己最好的結果。它不僅僅考慮了自己方下棋的利益，同時還將對手可能會下的位置與利益都考慮進去，避免了可能會出現的當下步路獲得大量棋子後，卻在對手的下一步後失去更多棋子的窘境。

但是這種演算法最大的缺點就在於，需要考慮的情況非常龐大，除了要耗費大量空間儲存外，還需要非常長的計算時間。因此如果可以結合Alpha-Beta剪枝法，將部分多餘確定非最佳解的節點樹剪去，可以省下許多儲存空間，同時也可以減輕電腦的計算負荷，加速AI運算出答案的時間。

三、人腦樣本

我們的AI是依據權重表，以及能翻轉的棋子總數來決定下一局應該下在哪一個位置，但這種行動模式很容易被熟悉權重表的人破解，所以我們組員到後來漸漸能猜出AI下一步要下在哪裡，甚至能把AI逼到角落，強迫它下在權重較低、不利於遊戲的位置。

聽完老師和助教的建議，我們認為可以用以下方法改進AI部分：

我們應該先徹底精熟黑白棋的規則，進行多場人對人的對弈之後，發展、總結出一套我們認為獲勝機率最高的戰略規則，將這套規則套用到AI身上，等於是把「人腦」當做樣本，再去設計AI的行動模式，讓它能更貼近我們的思考方式，對時時都在變動的棋局做出更有彈性、以設計大局為優先的判斷，而非只會依照不變的權重表行動。

**肆、改進**

一、排行榜

我知道，我們最終項目的最終結果遠非「完美」。我們有很多需要改進的地方，例如:可以通過顯示每位玩家的排名列表來完成其中一項改進。我們提出這個想法是因為我們在遊戲開始之前，就輸入玩家的名字，並在遊戲結束之後，計算了他們的得分，最後的頁面中可以呈現誰贏得了比賽，因此，為了改善用戶玩我們的遊戲的體驗，我們認為可以在遊戲中顯示每位玩家的排名和得分，並可以通過獲取每位玩家的得分，進行比較來顯示等級，然後使用#include<graphics.h>中的outtextx()函式，放入ShowGameWinMsg()中，在遊戲結束時顯示。

二、遊戲結束時的畫面顯示

可以完成的另一項改進是將遊戲結束時的「玩家1/玩家2贏得遊戲」更改為「（玩家1名稱）贏得遊戲」，可以通過用全局字串player1name[]或player2name[]替換“玩家一”或“玩家二”來完成，此替換可以在ShowGameWinMsg()內部完成。

**伍、心得**

B084020016陳育綺:

這次的專案有別於上學期程式設計課程的專案，當時從零開始設計出一個遊戲，因此在想法、概念及實作上比較容易結合。這次的專案則是在一開始先給我們初始的程式碼，由我們負責部分功能，這個部分在一開始給了我們很大的挑戰。由於前置的程式碼並不是用我們自己的思維去寫，在每一次的修改或是新增其他功能的時候，都要花很多時間先去釐清前面程式的來龍去脈，邏輯上經常會很難對起來。不過相對地，在後續慢慢熟悉整個程式之後，除了實作更上手，也在過程中體驗到別人用不同的切入點在思考整個邏輯的過程。

很高興在這次的專案中與同組的組員能夠實現有效溝通並且按照時間表逐漸完成成品，在報告結束後也聽聞其他組別的分工糾紛，對於此我特別感謝我的組員能夠對小組有所貢獻。我在這次的專案中學習到如何將事情做有效率的安排，也從與組員之間的討論過程中學到理性溝通並傾聽他人的意見，這對我來說是很大的收穫。

對於結果，雖然或許不像其他有些組別做了精緻的介面，AI也有蠻大的進步空間，但我認為以我們組現階段各方面例如能力、時間等考量下，已經可以算是盡力完成了，當然若有機會，還是希望能把老師、助教，以及觀摩後的自我檢討多多改進，讓小組的作品可以更好!

B084020023朱曼華:

在製作專案的過程中，我發現「先思考清楚再行動」是一件非常重要的事。我們小組起初對如何製作AI毫無頭緒，只好先嘗試讀懂助教給的程式碼，有了對程式碼的基本理解後，我才意識到我們並不用從零開始寫AI，其實助教已經給我們許多便利的工具了。運用了幾個助教已經寫好的函式及結構，例如計算能翻轉棋子總數的reverse，以及指向AI要下子的位置的focusPtr等等，我們AI的easy mode便完成了。

之後我們開始著手AI的hard mode，嘗試在程式中加入權重表，結果程式碼變得密密麻麻，我自己都看不太懂，還出現了好多bug。我們小組花了大量的時間debug，解決掉所有bug雖然非常有成就感，但我也意識到有些bug其實是因為我的急躁造成的。有時，我在腦中有了一點想法，沒有讓它慢慢沉澱醞釀便急忙打成程式碼，結果造成了許多邏輯錯誤，也使我的思緒變得越來越混亂，還要浪費兩三個小時debug。若是我能先思考清楚再行動，一定能省下更多時間，也能更有效率地製作專案。

B084020032王郁文:

製作遊戲和平時的實習作業不同，程式檔中包含多個函式，讓全域和區域變數的分別更加重要，也引入多個函式庫工具，加強自行搜尋網路資源並自學的能力，專案較長的篇幅更讓我經常修改了下面程式碼，卻忘記上面，在這次實作中，最為精進的技術是使用「Ctrl + F」，解決大海撈針的問題。

此次專案本組遇到最大的困難是，我們都對棋藝遊戲一竅不通。上網找遊戲規則和攻略，甚至不知道還有MINIMAX的使用方式，利用臨陣磨槍的知識去實施AI程式，導致AI也和設計師一樣是門外漢。

本來以為我們已經盡力，但在觀賞各組的期末展示後，發現本組缺少的是「創意」，太過於鑽研如何將黑白棋打造得如同已有的電腦遊戲，缺少自我發展的能力，事實上重要的不是再造已有資源，而是進步與創新，才有機會對科技的向前發展有所貢獻。

B084020034謝旻臻:

從最一開始聽到我們的期末專案要做遊戲的AI時，可以說是完全傻眼，畢竟以前從來沒有接觸過這類型專案的製作，再加上我們從大一到現在的lab，大部分都是以完成題目要求為主，幾乎沒有嘗試過把這些我們學到的演算法，實際運用在其他地方。

但是在經過網路上查資料、各方詢問之後，漸漸發現似乎AI的實作也不是那麼遙不可及，雖然在製作的過程中遇到了各式各樣的阻礙，還有在debug過程中不停出現新bug的崩潰輪迴，但是在一個個障礙以及bug被我們慢慢解決的當下，真的是非常有成就感的一件事情。

B084020043蔡昀芳:

透過這次的專案，我學到了如何把所學的知識融入到實作中，儘管在製作過程出現許多困難，也遇到許多沒見過的程式，但這反而可以讓我更加清楚地明白我還有許多地方需要學習，並且妥善地運用網路上的資訊來幫助我更好地了解程式的運作模式，學到了更多不一樣的程式。雖然這次的專案並不是非常完善，還有一些地方需要更進一步的改善，例如:可以把權重表從原本的靜態改成動態，這些都是我可以再去學習的地方，聽完同學們的報告、老師及助教的回饋後，我也更加了解有甚麼是可以改善的，有哪些是可以加強的。雖然這些都是平常在課堂上較少接觸到的範圍，不過卻可以藉由這次的專案讓我們接觸到，讓我們可以學以致用，這也是我在此次專案中學到最寶貴的經驗。

B084020048許映婷:

我們很努力去網上找不同的資料來完成這個專題。例如:怎樣去改背景顏色、做遊戲的列和行、黑白棋的顏色、玩家一跟玩家二分數的運作、還有加上按某個按鍵就會關掉和從新開始到原來的程式、最後的也需要考慮音樂的部分。說實話，我以前從來不知道c語言會放音樂在裡面，做了這個專題才想到要放不同的音樂在程式碼裡面讓玩家玩的時候會覺得這個遊戲很好玩，我們特地在網路上查了很多資料，最後決定在每個畫面放不同的音樂。例如:一開始的音樂及贏家和輸家都有自己的音樂。除了在上面加上很多功能，到最後我們還加上了一個能輸入玩家一跟玩家二的名字，讓玩家對玩家玩的時候他們會分的清楚到底是誰在玩那個遊戲。

B084020051丁海倫:

我相信我們小組學到了很多東西，通過這個項目，老師培訓了我們如何在小組中工作，我們學習到如何管理時間和進行協作、如何將每個人的想法結合起來以創造價值，這是「團隊合作」應有的典範，最終，我們也做到了!找到時間開會和討論的時間是我們進行此項目的過程中遇到的最大障礙之一，因為小組中有7個人，每個人的時間表都有所不同，因此我們通過花很多時間解決這個問題，以便我們可以一起討論。我遇到的另一個障礙是，這麼多人編寫程式，而我卻很難閱讀別人編寫的程式，畢竟每個人都有屬於自己編寫程式的方法，但這是一次很棒的經歷，因為即使每個人都很忙，但大家都還是對這個專案非常負責並積極的做出貢獻。