計算機網路 Computer Network（CE3007A）

Socket Programming 作業 - 五子棋

報告人：資工三 B　110403548　盧韋仲

- **程式執行截圖**

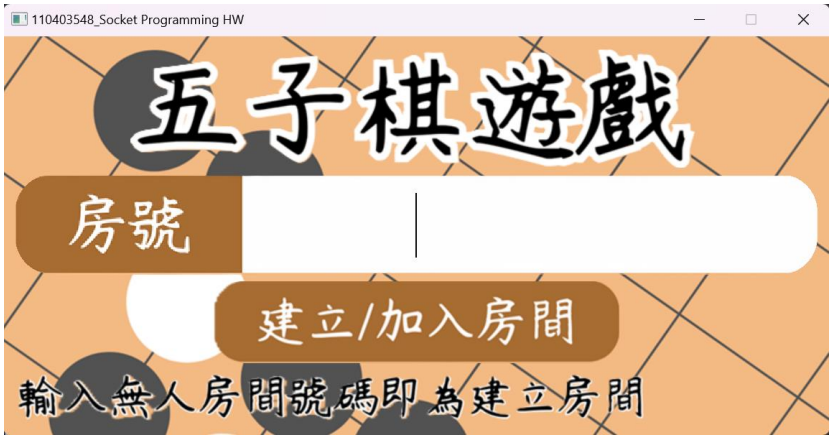
1. Server 端

```
user@LAPTOP-C9S3UPI9  D:  Programming  SocketProgrammingHomework  master
❯ python .\server.py
Enter 'stop' to terminate the server:
[*] Listening on 127.0.0.1:8000
[*] Accepted connection from: ('127.0.0.1', 55847)
[*] Accepted connection from: ('127.0.0.1', 55858)
A socket has left the room. Socket ID: 480
A socket has left the room. Socket ID: 452
stop
Server terminated.
```

Server 端可能之執行過程與其說明：

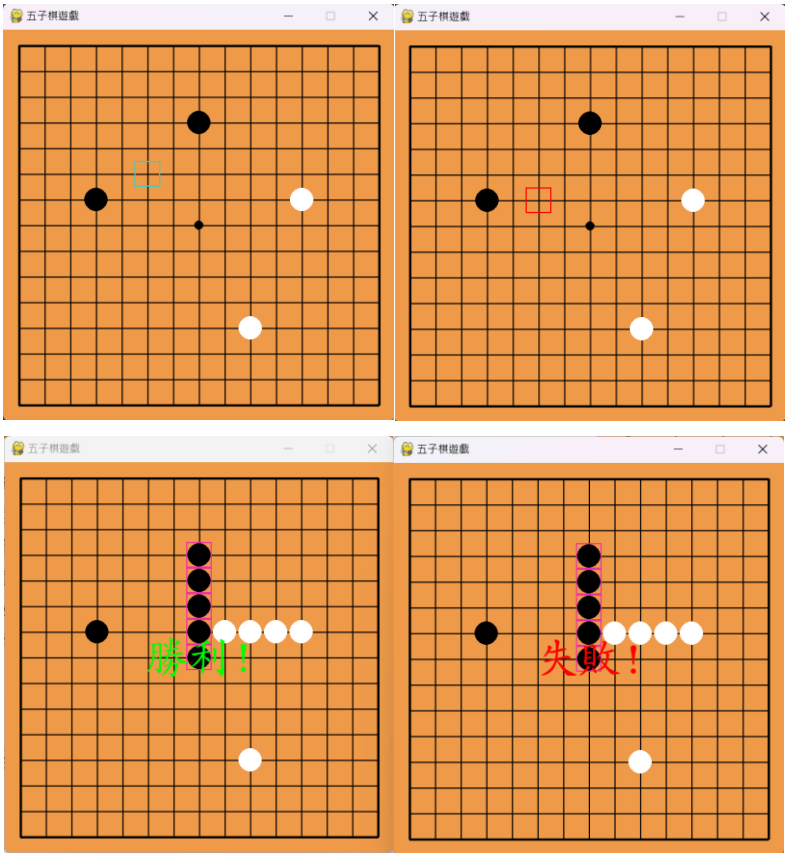
- 啟動後會顯示「Enter 'stop' to terminate the server:」的提示字。
- 顯示目前監聽的 IP 與 port。
  如：[*] Listening on 127.0.0.1:8000。
- 當 Client 端連線後，會顯示連線的 IP 和 Port。
  如：[*] Accepted connection from: ('127.0.0.1', 55858)。
- 當 Client 端因各種原因斷線時，會顯示其狀態。
  如：A socket has left the room. Socket ID: 480。
- 當使用者輸入 stop 後，會顯示 Server terminated，並關閉 server 端。

2. Client 端



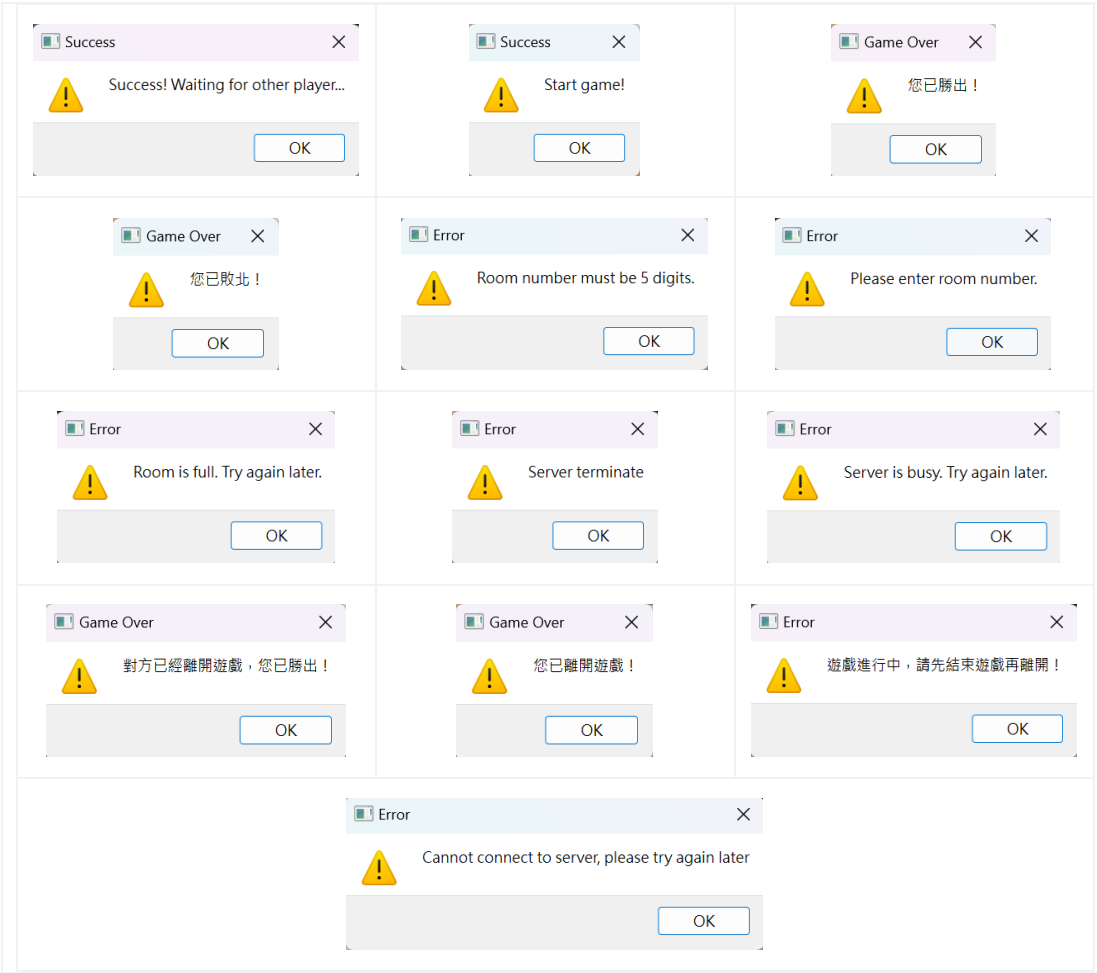
Client 端初始畫面與其說明：

- 中間的輸入框能夠輸入房間編號，並有以下規定（不符者會提示錯誤）：
  1. 只能輸入數字
  2. 房間編號必須剛好為五位數字
  3. 輸入無人的房間代表建立房間，輸入有人的房間代表加入房間
  4. 每個房間只能有兩個連線
- 輸入房間編號後點擊「建立／加入房間」按鈕即開始嘗試建立連線。
- 嘗試建立連線與遊戲進行中，按鈕會變得不可點選，以灰色的狀態顯示。
- 畫面可做簡易的放大縮小，但會受限於背景圖片的大小。



Client 遊戲執行畫面與其說明：

- 棋盤大小為 15x15 格。
- 圖形中央的黑點是用以定位的位置。
- 黑棋先下，白棋後下（建立房間者為黑子；加入房間者為白子）。
- 定位框的位置會跟隨滑鼠游標位置，即為點擊後的下棋位置。
- 輪到我方的回合會顯示藍色定位框（如左圖），對方回合則顯示紅色定位框（如右圖）。
- 遊戲結束將顯示勝利條件的位置（以粉紅色圈選），並出現「勝利!」或「失敗!」的字樣，同時畫面停止，無法點擊。（需點擊提示框以關閉視窗）



Client 端可能出現的提示框與其對應的狀態（由左而右，由上而下）：

- 建立房間並等候其他玩家加入房間。
- 加入已經建立的房間，即將開始遊戲。
- 遊戲勝出，點擊 OK 將關閉遊戲視窗。
- 遊戲敗北，點擊 OK 將關閉遊戲視窗。
- 房間編號輸入小於五位數字。
- 未輸入房間編號。
- 房間已滿，無法加入。
- Server 於等待時間關閉。
- Server 供應人數達到設定之上限（即 Thread 上限，將在後續說明）。
- 對方關閉遊戲視窗，或遊戲過程中 Server 關閉。
- 自身關閉遊戲視窗。
- 遊戲進行中關閉主畫面，將不予關閉，需先關閉遊戲。
- Server 未提供服務或無法建立連線。

3. 操作步驟

- Python 的安裝與環境變數等的設定。
- 完成相依套件的安裝（可參考後面檔案功能劃分區塊的指令）。
- 開啟一 terminal 並移動到所在資料夾並下`python .\server.py`以啟用 Server。
- 另開一 terminal 並移動到 Client 資料夾並下`python .\Page.py`以啟用 Client。
- 其餘操作與前述介紹相同。

• 程式執行說明/程式碼介紹

1. 函式庫使用

- **PyQt6：**
  圖形化使用者介面製作、控制與 QThread。
- **sys：**
  控制系統關閉（關閉視窗即關閉程式）。
- **NumPy：**
  矩陣的控制；即棋盤內容的紀錄與運算。
- **pygame：**
  五子棋遊戲介面製作與控制。
- **socket：**
  Socket 的連線、接收、傳送與控制等。
- **os：**
  設定環境變數，以控制 pygame 在 import 時不顯示提示文字。
- **threading：**
  設定與控制與 PyQt6 無關的多執行序。

2. 檔案功能劃分

- **requirements.txt：**
  紀錄 pip list，以方便使用 `pip install -r <path>` 來一次安裝所有相依套件。
- **server.py：**
  Server 端的程式碼。

- Client 資料夾
  - Resources 資料夾
    Client 端的資源檔，內含初始畫面所需的圖檔以及 pygame 所需的字體（標楷體.ttf）。
  - UI.ui
    由 Qt Designer 設計完儲存而成的 .ui 檔案。
  - UI.py：
    由 UI.ui 轉換而成的程式碼，負責畫面外觀的產生。
  - Page.py：
    Client 端的程式進入點，負責 PyQt 的畫面，包含讀入 UI.py 的外觀元件（QWidget）、設定控制介面事件、產生提示框、判斷輸入、建立連線等。
  - Game.py：
    五子棋遊戲的程式碼，並使用 pygame 實現遊戲畫面的顯示、遊戲之間的溝通與畫面控制等。同時使用了多執行緒來實現遊戲更新的非同步處理，以確保畫面的即時性
  - Background_rc.py：
    由 Resources 中的 background.qrc 轉換而成的程式碼，內容為資源包中圖片的 Pixmap，即基於 ASCII 編碼的圖像格式。
  - Thread.py：
    包含了兩個自定義的 QThread Class，分別是 GameThread 和 WaitingThread，以確保在執行不同任務時，畫面仍可保持響應狀態。並透過 pyqtSignal 提供訊號以在主程式中處理對應的事件。

3. 程式碼說明

由於程式中程式碼眾多且繁雜，故不在此提供程式碼截圖，若有需要請見所圖供之程式碼。同時在此不介紹自動產生之 UI.py、background_rc.py 以及資源控制的 qrc 檔案。

- server.py
  - rooms
    為一 Python 中的 Dictionary，負責將房間編號對應到所連接的 socket list（最多兩個）。
    如：10000 對應到[socket1, socket2]。

  - ADDR
    儲存連線的 IP，預設為 127.0.0.1。

- backlog
  儲存 socket 的連線數，預設為 10。

- PORT
  儲存連線的 Port Number，預設為 8000。

- MAX_THREADS
  儲存最大的執行緒數量，預設為 10。

- terminate_server
  儲存控制是否停止 Server 端的 Flag。

- if __name__ == "__main__"
  程式進入點，此區的程式碼會在直接執行這個檔案時執行，其中 main_thread 是主要的執行緒，負責執行 main 函式，同時將其 deamon 設為 True 表當此程式退出時，該執行續也會隨之終止。此時會進入無窮迴圈，輸出提示文字並等待使用者輸入 stop 字樣（以.lower 確保大小寫皆可成功）。當收到 stop 或 KeyboardInterrupt（如鍵盤強制終止等），將 terminate_server 變數設為 True，並輸出終止的提示字。

- main()
  負責 Server 的主要程式邏輯。會建立一個基於 TCP socket 的 Server，並由前述的變數綁定 IP、port 並監聽連線。當街收到新的連線時，會先取得 Client 傳來的房間編號，並至 rooms 檢查房間是否存在，若不存在則建立該房間。完成後，檢查房間是否滿二人，即房間編號對應到的 list 是否有存有兩個 socket，若已滿則拒絕連線，回傳「Room is full. Try again later.」；若未滿則將其加入對應的 list 中，並依據人數決定發送是否需等待其他玩家的訊息（需等待則發送「Success! Waiting for other player...」；不需則對等待方與 Client 二者發送「Start game!」），並以 handle_client 函式（下述）建立新的執行緒處理該 Client 的通訊，若執行緒已達到所設定的最大值，則輸出對應的提示並回傳「Server is busy. Try again later.」。以上任務持續直到 terminate_server 被設定為 False。當設定為 False 時，對所有以建立的連線傳送「Server terminate」並關閉所有連線與本身的 socket。

- handle_client(client_socket, room_number)
  負責處理與Client端以及Client之間的通訊，會不斷接收來自該client_socket的訊息並轉發給同一房間中的其他玩家。同時處理許多例外狀況，如：接收的訊息為空時斷開；使用者因斷線等因素離開房間時要輸出提示並將其移出list等。跳出迴圈後，將把連線斷開並從房間的list中移除該玩家。

- Page.py
  - ADDR
    儲存連線的IP，預設為127.0.0.1。

  - PORT
    儲存連線的Port Number，預設為8000。

  - Page類別（繼承QWidget，負責應用程式的主畫面）
    - __init__
      定義物件初始化的行為，包含:呼叫繼承之__init__方法、綁定從UI.py匯入的畫面、設置UI、設定視窗上元件對應的控制函式（呼叫set_Control），並顯示視窗。

    - closeEvent
      負責處理視窗的關閉事件，會檢查是否有遊戲正在進行中（藉由檢查是否有game以及game的game_running屬性），若是則顯示錯誤提示並忽略該關閉事件；若否則執行該關閉事件。

    - set_Control
      負責設定畫面上UI元件的事件連接，包含：將按鈕的點擊綁定至JoinGame方法；將輸入框改變（textChange，即使用者輸入）綁定至inputRoomNumber方法。

    - inputRoomNumber
      負責檢查房間編號的輸入，會限制輸入格式(須為全數字，否則將非數字的部分移除）和長度（上限為5位數字，多餘的部分將移除）。

- getRoomNumber
  取得並回傳輸入框中的房間編號。

- checkRoomNumber
  負責檢查房間編號的合法性，即藉由 getRoomNumber 的方法取得輸入之房間編號，並檢查是否為空值、是否全為數字、長度是否為 5（可能大於或小於），若有以上情況，顯示對應的提示框表示錯誤並回傳 0；否則將回傳 1。

- JoinGame
  處理加入遊戲的事件。首先會藉由 checkRoomNumber 檢查房間編號的合法性，若合法則嘗試連接到 Server、發送房間編號、接收 Server 回應並將回應作為參數呼叫 handleResponse 方法；若連線、傳送或接收過程中失敗，則顯示內容為「Cannot connect to server, please try again later」的提示框。

- handleResponse
  負責根據所傳入之 Server 的回應執行對應的處理，包含：房間已滿、Server 忙碌中（執行緒達到設定上限）、成功建立房間、成功開始遊戲、Server 關閉、接收到其他訊息等。除了顯示對應的提示框外，其中成功建立房間（回應為「Success! Waiting for other player...」）時，會開啟一個定義於 Thread.py 的 WaitingThread 來接收來自 Server 的進一步訊息，並將該 Thread 的結束訊號 responseReceived 綁定至 doneWaiting 方法，同時將畫面上的按鈕取消功能，以避免多次或重複連線的狀況。另外，成功開始遊戲時會建立定義於 Game.py 的 Game 物件，並將其傳遞給定義於 Thread.py 的 GameThread，並將該 Thread 的結束訊號 gameover 綁定至 game_over 方法，同時將畫面上的按鈕取消功能，以避免在已經有遊戲進行中，同一 Client 建立其餘連線的例外情況。

- doneWaiting

  負責當 WaitingThread 結束並發送 responseReceived 的 pyqtSignal 時，依據得到的回應處理執行緒完成後的邏輯，即當收到「Server terminate」時，代表等候其他玩家加入時 Server 關閉，此時應顯示提示框並恢復畫面上的按鈕之功能以等待玩家再次嘗試連線；若收到其餘訊息，將退出 waitingThread，同時同之前成功開始遊戲的情況，即建立定義於 Game.py 的 Game 物件，並將其傳遞給定義於 Thread.py 的 GameThread，並將該 Thread 的結束訊號 gameover 綁定至 game_over 方法，同時將畫面上的按鈕取消功能，以避免在已經有遊戲進行中，同一 Client 建立其餘連線的例外情況。

- game_over

  負責處理遊戲結束後的提示訊息和清理邏輯，即嘗試讀取是否為主動關閉遊戲（game.quitFlag == True）、是否自身為勝利方、是否自身為敗北方並顯示對應的提示框。例外情況為對方斷線（或 Server 端於遊戲過程中終止），此時一律顯示「對方已經離開遊戲，您已勝出！」的提示框。最終會恢復畫面上的按鈕之功能，同時執行 game 的 game_over 方法以完成 pygame 執行的清理。

- showPopup

  負責顯示提示框，即依據傳入參數（title、content）設定提示框的標題與內容、設定顯示設定等，最終顯示該提示框。

- if __name__ == "__main__"

  程式進入點，此區的程式碼會在直接執行這個檔案時執行。執行後會負責建立一個名為 app 的 QApplication 和名為 login_window 的 Page，同時設定該視窗的標題為「110403548_Socket Programming HW」，並藉由 app.exec() 啟動該 PyQt6 應用程式的主事件迴圈，同時包入 sys.exit() 中，表示在主事件迴圈結束後會通知 Python interpreter 退出、回傳 return value 並關閉該程式，以達到正常結束應用程式的效果。

- Game.py
  - os.environ['PYGAME_HIDE_SUPPORT_PROMPT'] = "True"
    負責設定執行時的環境變數，將其設為 True，以防止 pygame 在 import 時出現如「pygame2.5.2(SDL2.28.3,Python 3.11.4) Hello from the pygame community.https://www.pygame.org /contribute.html」的歡迎訊息。

  - Game 類別
    - __init__
      定義物件初始化的行為，包含初始化遊戲，建立 pygame 視窗並設定顯示的標題，設定物件屬性(如：顏色、回合、目前輪到的玩家等)。

    - start_game
      負責啟動遊戲，會在 Page 所建立的 GameThread 執行時被呼叫。會透過 threading 中的 Thread（因非 PyQt6 的畫面，故不同於 GameThread 與 WaitingThread 的 QThread）建立一個新的執行緒執行 receive_game_updates 方法。建立並啟用完畢後，呼叫 run_game_loop 方法。

    - stop_game
      負責停止遊戲，即將 game_running 設為 False 以影響 run_game_loop 和 receive_game_updates 方法的執行。

    - check_win
      負責檢查遊戲是否有五子連線（即一方勝利、一方敗北）的情況，並回傳贏家的編號和連線位置，以利後面畫面的標示與顯示文字的判別。判別過程即透過一 NumPy 的陣列（概念上同 list)，來表示棋盤各格的位置與狀態，一一走訪傳入的棋盤狀態（over_pos）並透過直向、橫向、斜左、斜右的判斷(即 for 迴圈)來決定並回傳是否有連線。其中回傳的第一個參數為哪一方獲勝（0 表無人獲勝；1 表黑子獲勝；2 表白子獲勝)，第二個參數為一 list 儲存所判別到連線的位置。

- find_pos
  負責依據所傳入之 x 和 y 值，計算並顯示所對應可以落子的位置，即回傳其對應到 x 和 y 值所處在的格子位置（過程中扣除邊框的畫素等）。

- check_over_pos
  負責檢查所傳入的 x 和 y 位置在目前的棋局狀態下（傳入的 over_pos）是否已經有落子。

- run_game_loop
  負責遊戲的主迴圈，其中主迴圈受 game_running 屬性所控制。迴圈中將處理使用者輸入（如關閉視窗或按下 Esc 時向所連線之 socket 傳送「GameOver.」並退出遊戲）、畫面渲染、遊戲邏輯（呼叫 check_win 方法，若有成功會從資源包中取用標楷體並顯示勝利或失敗的字樣）、獲取滑鼠游標位置（呼叫 pygame 中 mouse 的 get_pos 方法，依據 check_over_pos 方法判斷是否落子和自身的 turn 與 current_player 屬性以顯示藍色或紅色的落子提示框）、滑鼠點擊事件（除落子外也需向所連線之 socket 傳送如「Move:100,200」的訊息）、畫面更新（以 Flag 和 tim 控制時間點即事件的時間間隔長短，以避免按住滑鼠或連續點即導致一次下多子的情況）。

- game_over
  負責處理遊戲結束，即將 game_running 設為 False 以影響 run_game_loop 和 receive_game_updates 方法的執行，同時釋放相應資源（關閉 socket 的連線）並關閉視窗。

- receive_game_updates
  負責接收與處理從 Server 接收的訊息，並產生對應的遊戲更新。如同 start_game 所述，會在一個獨立的 Thread 中進行，以維持畫面的持續更新、避免畫面停止響應。

- Thread.py
  - GameThread

    __init__表初始化時，將接收一個 Game class（定義於 Game.py）作為參數。run 為執行緒啟動時的動作，會呼叫 Game class 的 start_game 方法，此時執行緒會進入該 pygame 中，直到遊戲結束才發送一個名為 gameover 的 pyqtSignal 給呼叫的 class（Page）接收。

  - WaitingThread

    __init__表初始化時，將接收一個 socket 作為參數並將變數 isRunning 設為 True（用以控制執行緒）。Run 方法為執行緒啟動時的動作，會持續監聽 Server 的回應，當收到 "Start game!" 時發送 responseReceived 訊號，並將 isRunning 設為 False；否則將回應發送到呼叫的 class（Page）接收（通常用以處理 Server 在等待的過程中終止的情況）。Quit 方法為停止執行緒時使用，會將 isRunning 設為 False 並呼叫其繼承的 quit 方法以終止執行緒。

作業要求完成度

1. 製作 TCP Socket 的程式
   - 以 Python 實作

2. 本實驗報告

3. 加分項目
   - GUI 介面
   - 多 Client 連接（Multithreading）
   - （Non-blocking Socket）由於執行過程皆為會以獨立的執行緒監聽，因此雖未設定為 Non-blocking 但仍可持續執行程式中的其他部分，達成 Non-blocking 的概念。
   - （功能完整與有創意無法自評）