一、簡介

我們的報告主題是 UNO，使用 C 實現了 UNO 的基本規則。玩家可以選擇要開房間、使用房號碼加入房間或是隨機匹配，也可以在局內使用聊天室聊天。

- 分工：
  111550002 林宜頡：server
  111550164 廖涵玉：client

- 開發與執行環境：WSL Ubuntu 22.04.3 LTS

二、研究方法與設計：

因為 UNO 為回合式的遊戲，不須傳輸大量訊息，因此我們使用 TCP 傳輸。
（即時的部分：玩家意外斷線、玩家在遊戲中投降（不用輪到他）、玩家喊UNO、抓別人沒有喊UNO）

Server 主要負責處理大局邏輯的部分，包括分配房間、更新房間成員的出入、洗牌與發牌、紀錄與控制遊戲進行的狀態，並需要將狀態更新轉告給所有房間內的成員，使所有人的資訊同步。

Client 主要負責處理玩家各自的動作以及遊戲畫面的呈現（ANSI escape sequence），包括及時顯示 Server 傳來的更新資訊、設計出讓使用者體驗良好的操作介面、在局中計時，以及最重要的檢查玩家的輸入是否合法，像是牌能不能出、名字有沒有非法字元等，這樣才能保證Server 不會因為非法輸入而出錯。

- 資料傳輸格式：
  （所有資訊都以空格隔開，每筆資料以 '\n' 結尾，方便拆解）

  Server -> Client：

1	接收到玩家第一次進入大廳的 request，要求玩家輸入暱稱
2	暱稱設定成功
3	暱稱已存在，暱稱設定不成功
4 token	成功進入大廳
5 房號 房主暱稱 其他成員的暱稱	成功進入房間
6 新成員暱稱	通知其他成員房間有新成員加入
7 成員暱稱	通知其他成員該成員離開

8 新房主暱稱	通知其他玩家房主離開，房主變更
9 所有成員暱稱	遊戲開始，通知所有成員出牌順序
10 牌堆最上一張牌 牌庫數量 輪轉方向 輪到的玩家暱稱 所有人的手牌數量 你的所有手牌	通知所有成員新回合開始與目前資訊
11 成員暱稱	通知所有成員這個成員亂喊 UNO
12 新牌	發牌給此玩家
13 被抓的成員暱稱 抓到的成員暱稱	剩一張牌的人被成功抓到沒喊 UNO
14 成員暱稱	通知所有玩家此成員喊到 UNO 了
15 贏家暱稱 贏家分數	遊戲結束
16	找不到房間
17	此房間人數已滿
18	此房間由遊戲中
19 亂喊成員暱稱 被喊成員暱稱	通知所有成員此人亂抓別人UNO
20	牌庫見底，將廢牌充新洗牌的通知
21 暱稱:message	聊天室新訊息

Client -> Server:

(token) 1	進大廳的請求(如果登入過了，附上第一次登入時server給的token)
2 暱稱	登入
3	開房間
4 房號	加入房間
5	隨機匹配
6	退出房間
7	遊戲開始(房主才能)
8	投降
9 card_id	出牌(一次一張)
10	要抽一張牌

11 0/1 時間	喊自己UNO
12 0/1 要抓的人的暱稱 時間	抓別人沒喊UNO
21 暱稱:message	要傳訊息到聊天室

- 例外狀況之分析與處理:
    1. 玩家輸入的暱稱不符合格式:Client 會告訴玩家暱稱不合法，不會傳送給 Server
    2. 玩家在不是自己的回合出牌、玩家出不符合遊戲規則的牌:Client 會通知玩家，不會傳送給 Server
    3. 玩家突然斷線:Server 會關閉該玩家的連線，並通知其他玩家，其他玩家仍可繼續遊戲
    4. 所有玩家都退出，只剩一人:直接判該玩家勝利

三、成果:

----------登入階段----------

登入成功

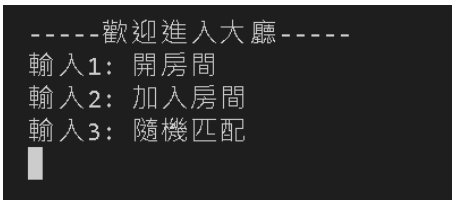
```
hanyu@hanyu:~/unpv13e/tcpcliserv$ make final_cli && ./final_cli 127.0.0.1
make: 'final_cli' is up to date.
連線中...
成功連線!登入階段，請輸入玩家名稱(限大小寫字母，限長10。
hanyu
登入中...玩家名稱：hanyu
登入成功！進入大廳中...
```

登入失敗

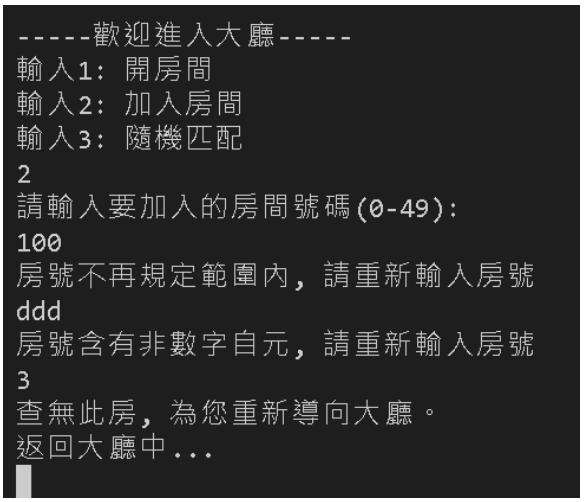
```
hanyu@hanyu:~/unpv13e/tcpcliserv$ make final_cli && ./final_cli 127.0.0.1
make: 'final_cli' is up to date.
連線中...
成功連線!登入階段，請輸入玩家名稱(限大小寫字母，限長10。
hanyu
登入中...玩家名稱：hanyu
登入失敗，玩家名稱已存在!返回登入階段...
登入階段，請輸入玩家名稱(限大小寫字母，限長10。
aaaaaaaaaaaaaaaaaaaaa
玩家名稱超過長度限制，請重新輸入玩家名稱
11111111111111111111111
玩家名稱不合法，請重新輸入玩家名稱
玩家名稱超過長度限制，請重新輸入玩家名稱
```

——----------大廳----------

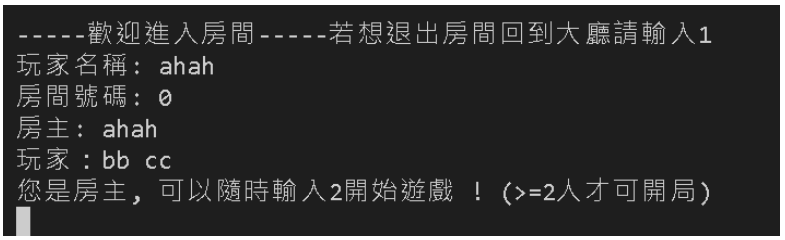
進大廳



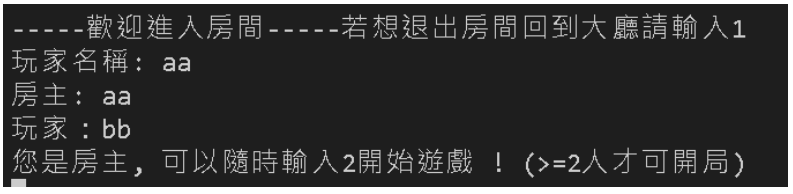
加房間失敗



開房間、加入房間成功



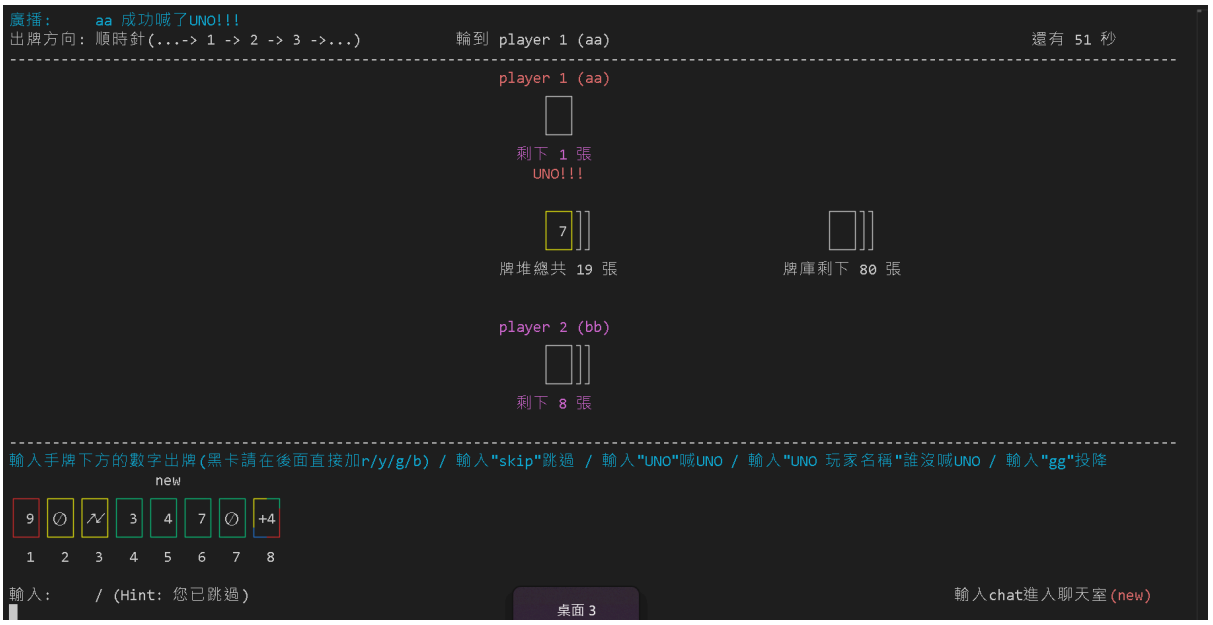
隨機匹配



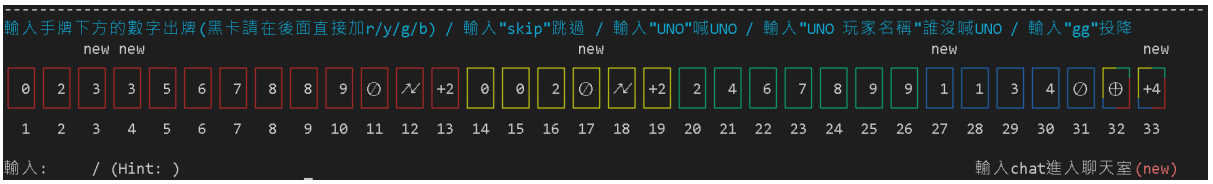
(都是房主決定何時開局) (2-10人) (房主退出房間也會重新指定房主)

----------遊戲----------

1) 廣播（有事件發生會即時更新）
2) 出牌方向、輪到誰（每回合由server告訴client）
3) 輪到的人會有”輪到你了”的提示，並且玩家名稱會變成紅色
4) 這回合倒數秒數（透過select timout = 1s達成）
5) 玩家在牌桌的分配（自己永遠是最下面那個，依照出排順序player1-n順時針分配）（喊UNO的人會有”UNO”在下方）（已離開的玩家會顯示已離開）
6) 桌上的最後一張牌和牌庫的牌（1張、2張和三張以上會分別畫出不同張數的牌堆）
7) 輸入提示（藍色那排）
8) 手牌區域（可以畫33張牌，超過會變成全部用文字印出來）（新的牌上面會有”new”）
9) 輸入和提示：無效輸入可能會有提示印出來（還沒輪到你、這張牌不能出.....）
10) 聊天室：有新訊息時會有”new”



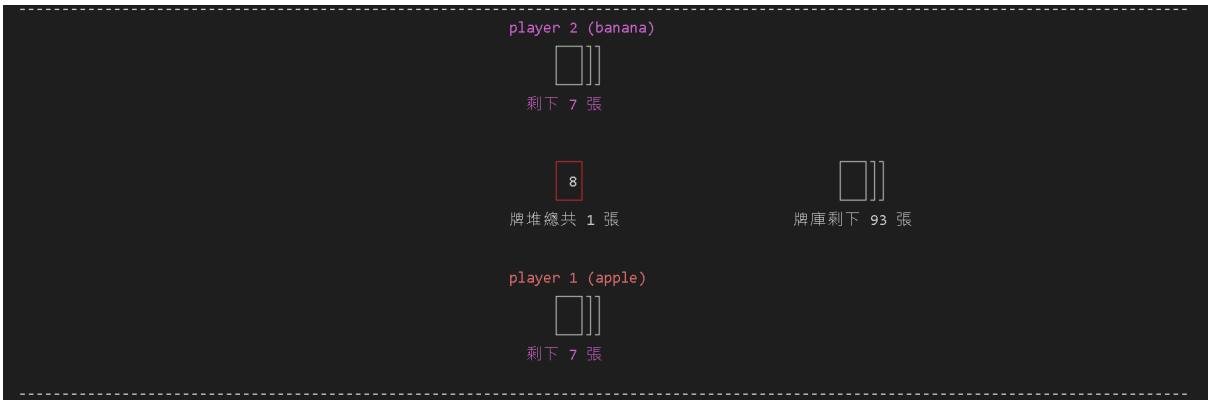
手牌區（最多畫出33張）



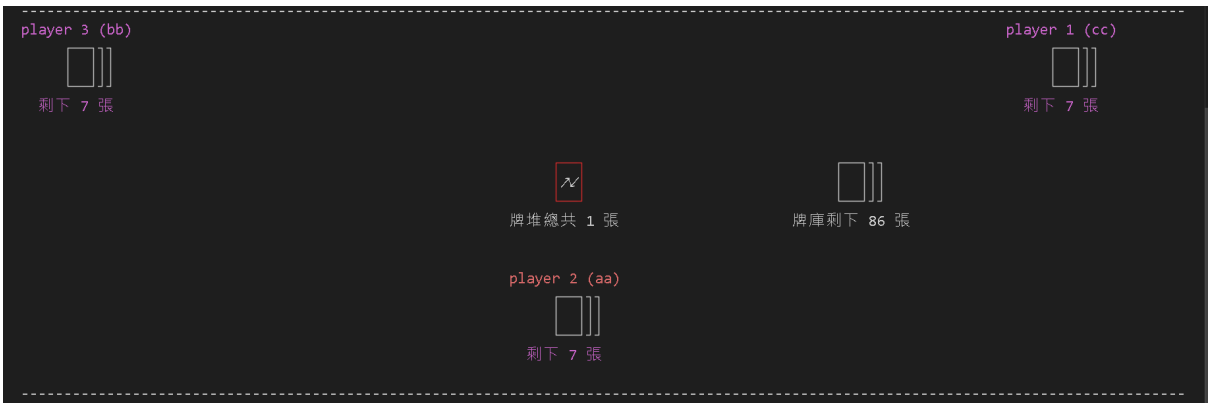
聊天室

投降

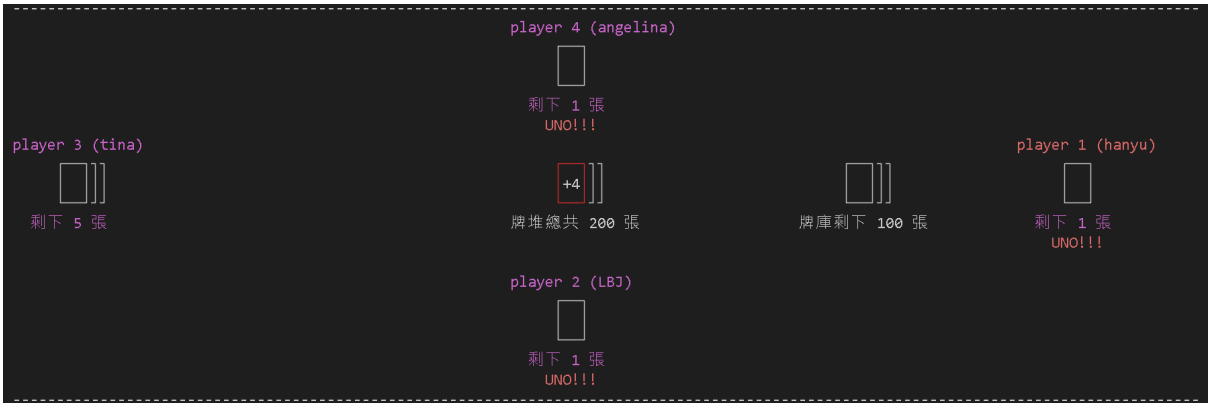
2人局



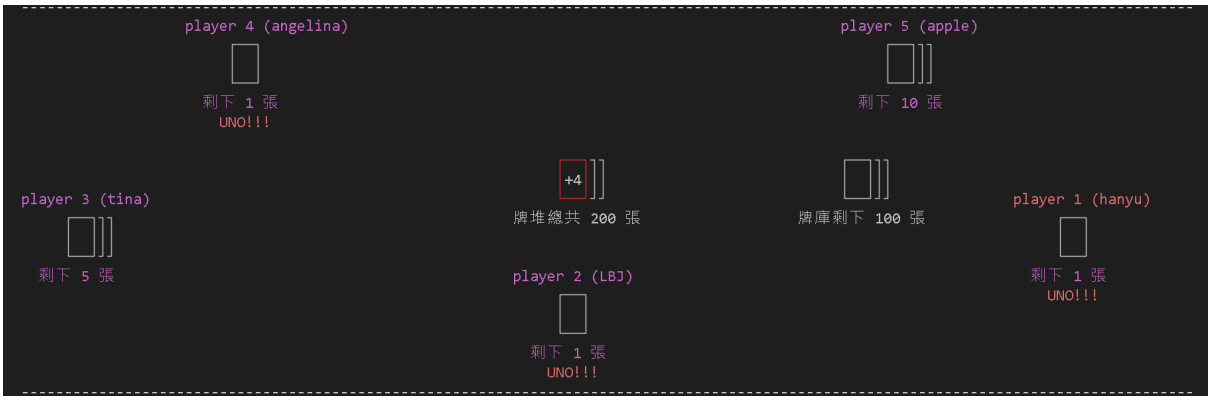
3人局



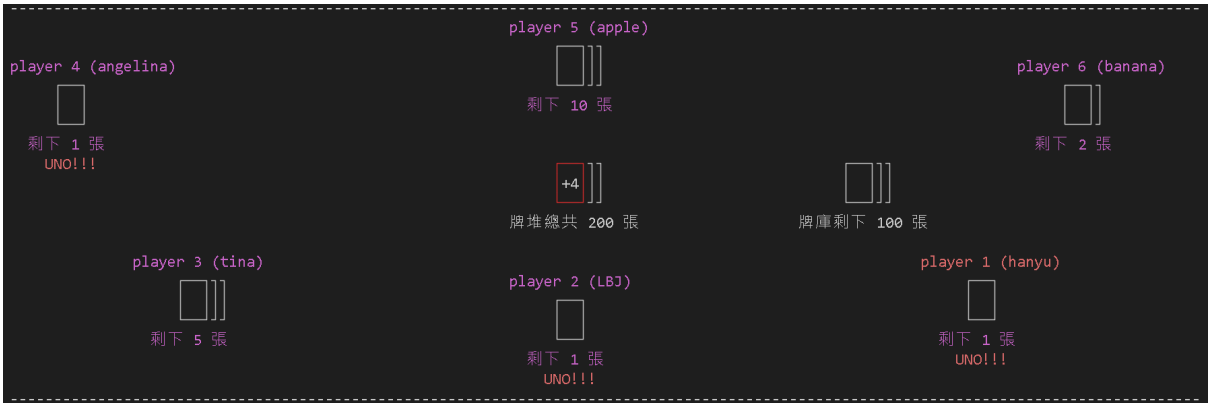
4人局



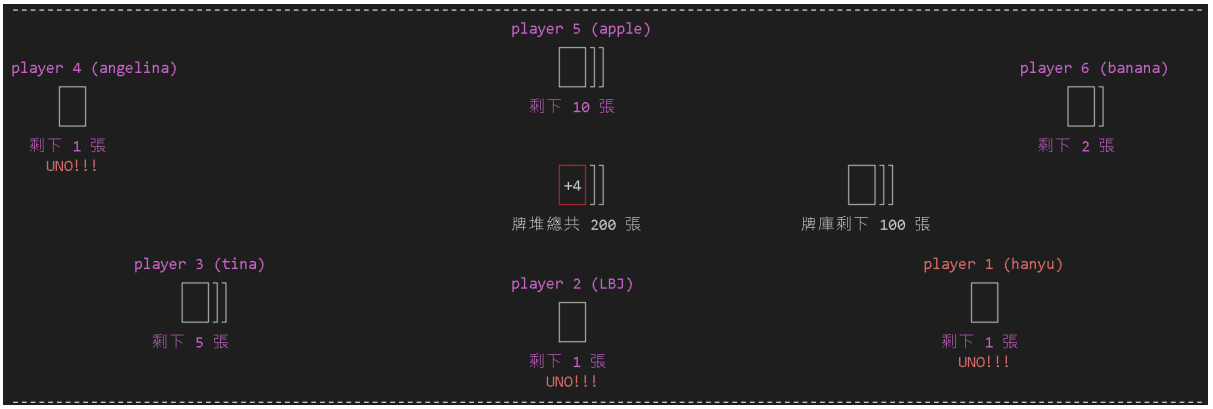
5人局


player 4 (angelina)
剩下 1 張
UNO!!!
player 5 (apple)
剩下 10 張
+4
牌堆總共 200 張
牌庫剩下 100 張
player 3 (tina)
剩下 5 張
player 1 (hanyu)
剩下 1 張
UNO!!!
player 2 (LBJ)
剩下 1 張
UNO!!!

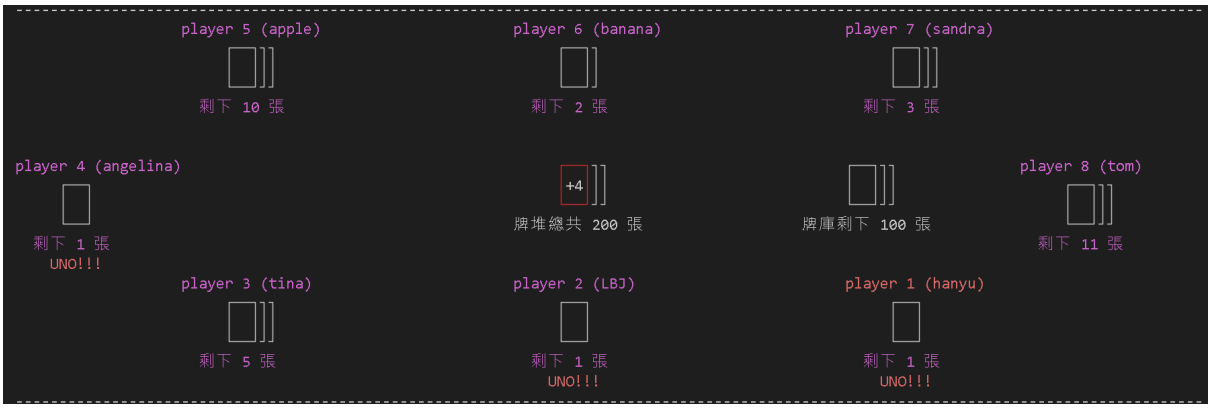
6人局


player 5 (apple)
剩下 10 張
player 4 (angelina)
剩下 1 張
UNO!!!
player 6 (banana)
剩下 2 張
+4
牌堆總共 200 張
牌庫剩下 100 張
player 3 (tina)
剩下 5 張
player 2 (LBJ)
剩下 1 張
UNO!!!
player 1 (hanyu)
剩下 1 張
UNO!!!

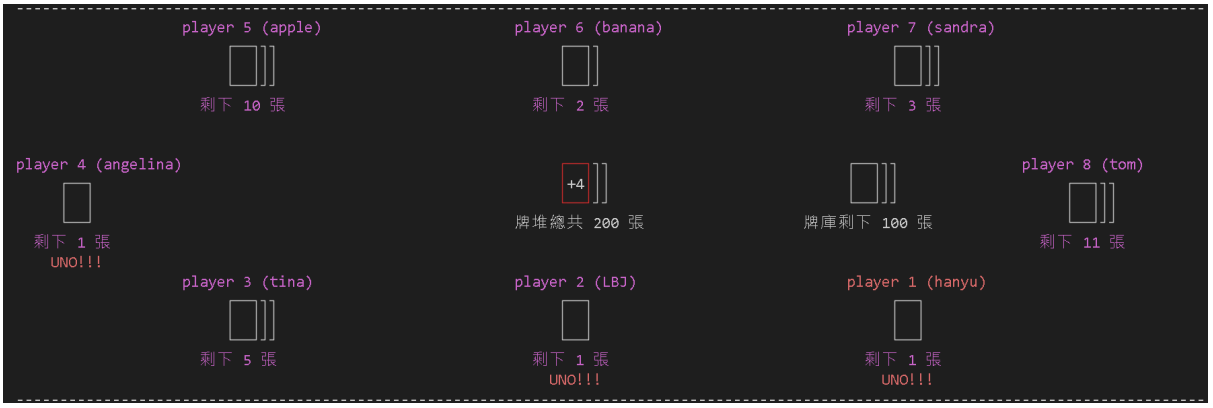
7人局


player 5 (apple)
剩下 10 張
player 4 (angelina)
剩下 1 張
UNO!!!
player 6 (banana)
剩下 2 張
+4
牌堆總共 200 張
牌庫剩下 100 張
player 3 (tina)
剩下 5 張
player 2 (LBJ)
剩下 1 張
UNO!!!
player 1 (hanyu)
剩下 1 張
UNO!!!

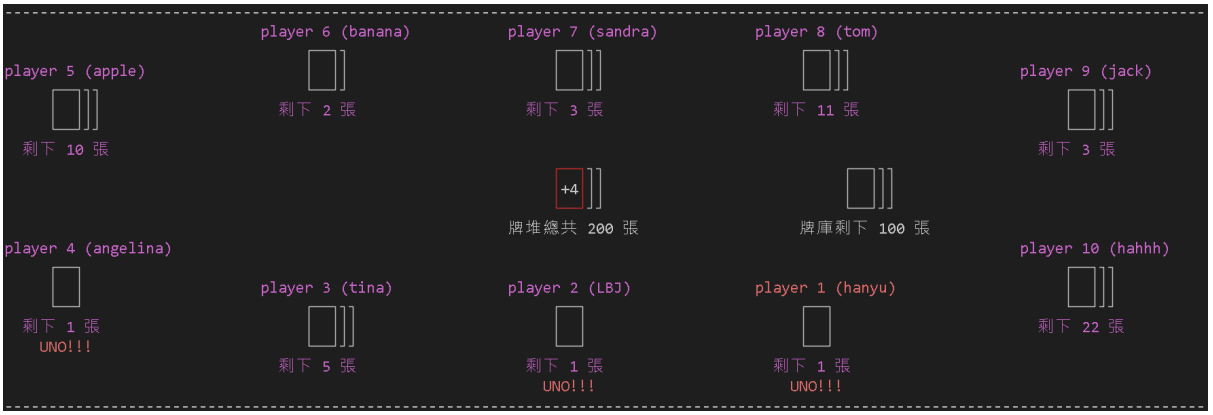
8人局



9人局



10人局



四、結論:

困難:

1. **select**:在剛寫到遊戲階段的時候，我們發現如果連續收到多則訊息，select 只會觸發一次把第一則讀出來，而即使後續幾則訊息都在 socket buffer 中（手動呼叫 readline 能夠順利讀取後續的訊息），select 也不會再觸發。後續經過不斷嘗

試，我們發現在需要連續傳訊息的情況時，在傳每則訊息之前加上 sleep(1)，把傳送時間隔開，就能正常接收訊息。由此我們歸納出一個結論：可能是由於環境問題，若是同一批送到 buffer 的資料，只會觸發 select 一次，後續就算資料沒有被一次讀取完成，還是有其他資料疑留在 buffer 中，仍然不會再觸發 select。

解決方法：每次select出來就一次讀掉buffer裡面的全部訊息（readline改成read），之後再用 ”\n” 拆成一行一行來處理。

2. **multi-threading (server)**：server 是使用 thread pool 來實現每個房間的處理，每個房間的 thread 都是由各自的變數（0 是不要執行，1 是執行）來控制要不要開始執行。但是中途遇到了明明已經調整變數到 1 了，thread 卻沒有成功啟動的問題。

   解決方法：後來在 main 與 thread 中都加上 “ __sync_synchronize(); “，強制內存同步，thread 才能順利執行啟動。（即使我們仍認為 thread 本身就不應該出現不能同步的問題。）

3. 輸入的游標 **(client)**：要每秒更新回合秒數(在螢幕右上方)，但是原本玩家輸入到一半的訊息就會被打斷，游標會跑來跑去。

   解決方法：先印出原本輸入時應該要在的位子，在每次要重新畫桌面的時候，把當前游標位子存起來，畫完再恢復（搭配fflush (stdout)），就可以解決了！

```
void game() {
    //收過9了
    printf("遊戲即將開始...\n");
    sleep(1);
    printf("\033[31;1H輸入: \n");
```

```
}
else {
    if (!dont_draw) {
        printf("\033[s");
        draw_table(n, player, playe
        if (renew) strcpy(new_card,
        renew = false;
        printf("\033[u");
        fflush(stdout);
    }
    dont_draw = false;
```

（每次重畫桌面時）

成果未來改進與延伸方向：

1. 寫 **GUI**：實作出能使用更直覺的互動方式的 GUI，例如用點擊的方式選擇要出的牌，讓使用者不必自行輸入指令，提升使用者體驗。
2. 外接資料庫紀錄玩家資料：可以新增登入系統，讓玩家自行決定暱稱跟密碼，並把這些紀錄在資料庫，讓玩家下次登入時仍能使用自己的帳號。資料庫中可以記錄玩家的積分，或許能在大廳中顯示全伺服器前五名玩家的分數，讓玩家從中獲得成就感。
3. 改成 **fork**：目前 server 處理各房間的方式是使用multi-thread，這讓 server 一次能處理的房間有限，或許可以改用 fork child 的方式增加房間數量的上限。
4. 讓遊戲結束後可以回到原本的房間：目前遊戲結束後會讓所有玩家回到大廳，改成遊戲結束後可以回到原本的房間可以讓使用者體驗更好。

心得：

- 廖涵玉:

  原本想要用GUI但是從來沒用過，怕會來不及生出來，又覺得用ascii畫也滿可愛的，所以就用ANSI escape sequence來畫了（主要是因為找到了禁止符號⃠跟萬用牌的符號⊕，還有迴轉的符號↗↙就覺得可以用ANSI escape sequence畫了）。

  收穫最大的應該是字元陣列的處理跟ANSI escape sequence移動游標，用酷酷顏色來畫桌面，還有要怎麼排放、整理想要呈現給玩家的東西。看畫出來的可愛卡片真的會很開心，讓漫長的debug過程變得比較不會生氣。

  這是我第一次寫這麼長的code，而且還要隨時跟隊友一起debug，不然如果兩個人進度差太多就會對不上。如果有一個人發現問題，另一個就會趕快衝過去一起檢查，要趕快把自己的bug改掉，是很有趣的體驗哈哈哈。

  幸好在期末考完之後有完整的時間可以來寫，我覺得一步一步寫出這個專題很有成就感，寫完之後也有跟室友一起玩，發現有超多bug是需要大家一起完才會出現的。

- 林宜頡:

  這次的作業剛好我跟隊友的相性蠻互補的，我是非常堅定想寫 server，而她是非常適合寫 client，畢竟她真的太有藝術細胞了。而過程中我也越來越確信了這點，我看著我隊友 client 中那坨讓人害怕的畫圖的 code，還有好看的畫面，就越來越覺得當初這樣分工真是太對了，我大概一輩子都刻不出這種東西吧！

  我覺得這次 final project 真的很有趣，而且是在期末之後，沒有其他事情需要煩惱，可以專心的寫。隊友也很棒，溝通的過程都很順利也很愉快！最後的成果我也很喜歡，大概是之後無聊會想拉朋友一起玩的程度吧。我們再 demo 前一天有邀請朋友們一起玩，喊 UNO 的環節是真的蠻刺激的哈哈。

  在寫 server 時實在是經過了好多波折（雖然很多時候都只是因為自己太笨）。一開始我是打算讓玩家開房間的時候就先 fork 出去，有新玩家想加入房間時，再通過 shared memory 將玩家的 connfd 送給那個房間的 child，還以此為基礎開心的寫了一整天。結果當然，在開始測試其他玩家要加入房間的時候出事了，又花了一整天 debug，才發現關鍵原因：connfd 只有創他的 process 能用啊！我居然忘了！當時隊友的 client 已經在寫遊戲中的部分了，我只能匆忙改成邏輯差不多的 thread，不過事後想想應該可以讓匹配階段用 thread，遊戲開始時再 fork 出去的。

參考文獻:

1. 老師的課堂 PPT

2. https://boardgamehot.com/uno-rule/
3. https://zh.wikipedia.org/zh-tw/UNO

附錄：

1. 本次作業的 code：
   https://github.com/yij10/UNO/tree/master