**Server程式碼&程式說明**

****

主宣告大多都擺在最上方，與Clinet端不同的是多宣告了資料庫用的容器負責裝玩家名稱遊玩時間與遊玩次數，以及開始程式時便設定port建立UDP socket，其他基本與Client端無異同。



設定好port後直接進入遊戲開使環節（迴圈），輸入題目的長度（因應題目要求故輸入0即跳出遊戲結束此程式）與題目並存下，當判斷到ans長度為1且字串為0時跳出迴圈，接著接收Client端送來的Client IP（傳封包用）。

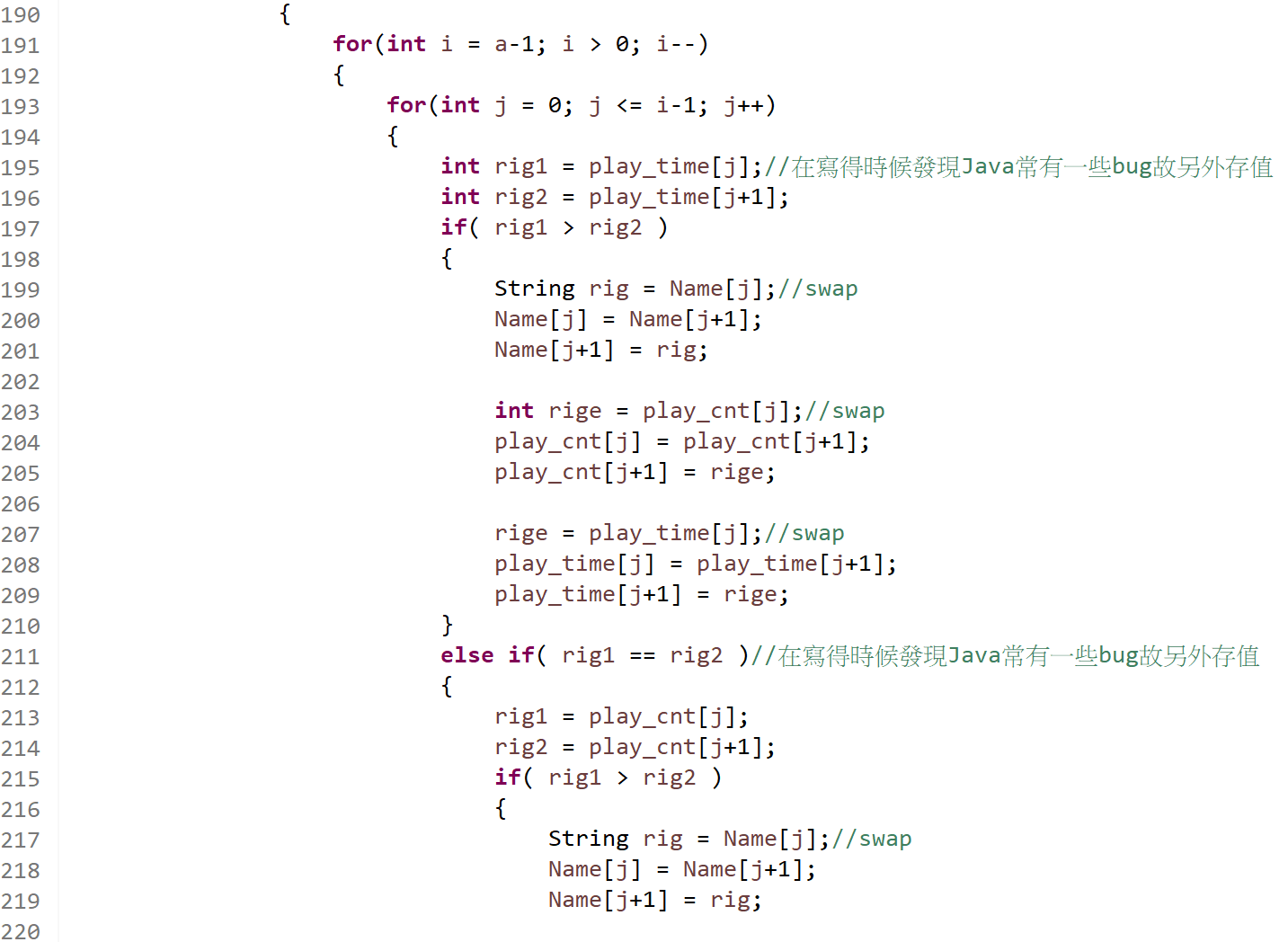




藉由剛收到的Client IP接著將題目長度送給Client端，而後進入判斷幾A幾B的迴圈重複著接收Client的猜測並與題目判斷後送回（此處送回\_A\_B字串以及A個數），直到Client端猜出正確答案，跳出迴圈後接收玩家名稱、猜測次數與遊玩時間（128-150列）。

先將本次結果寫入rank.txt檔立（後面的true是指有檔案就直接寫入，沒有則創一個寫），再第最後一列的地方我加入了？讓後面的讀取去判斷是不是讀到最後一個（之所以這樣想是因為無法使用是否為空字串去判斷該不該繼續讀入）。先將排名初始化後開始使用Scanner API去讀取txt檔內的數據並同時記錄總共有幾筆資料。Catch內的輸出則是在除錯時使用的。



接著開始讀取資料庫內的所有資料分別存在開頭所宣告的array中，後續判斷資料個數是否不為一，若是唯一就代表他是第一名（因為內部無資料），此處採用bubble sort根據猜測次數與遊玩時間去比名次（比較遊玩時間，較長者就往後擺，若次數相同則將猜測次數多的往後擺），另外使用swap函式時也會出問題故使用此方法慢慢交換。



根據排名排好後再找出本次遊玩紀錄存在陣列第幾個（因陣列重零開始故正式排名的rank要加1），首先我會用到一個布林值判斷有沒有在此陣列，以及本次遊玩的資訊，首先先找名字長度相同的，接著找每一個字元是否相同，然後時間跟次數是否相同，只要有一項不符就繼續找，直到找到正確的紀錄。



此處我先將排完排名的資料以（新寫）的方式寫入到rank.txt檔裡主要是為了消除原檔案的？字元，而後為了滿足題目要求用粗體字顯示本次遊玩紀錄我另外寫了html檔，for裡面丟陣列裡的資訊，按照排名由上往下排，若找到本次排名則以h4的head分開寫（標示粗體字）

****

最後傳排名給Client端後接收是否繼續遊玩的資訊繼續玩就回到開頭的迴圈重新出題，不玩就直接跳出迴圈結束程式。

**流程圖(Server)**



心得

此次是我第一次寫連線類型的程式，也因為此次經驗使我更加深理解傳送資訊的實際狀況，包括傳輸封包的先後順序，實際傳送與接收的port等，而寫的時間也花了不少，至少再除錯上就下了不少功夫，從port的接收與傳送必須不同相異到Java類型上轉換的問題等，要實際寫才會知道，尤其類型的轉換經常會有讀不出指定字串導致判斷異常，再加上排名系統我是使用txt檔去寫又有一些是要實際執行才有辦法找到的問題，像是找不到=NULL的狀況導致無法跳出迴圈等等最後總共花了我兩天左右的時間才全部完成，雖說有些判斷可能有限制，像是猜測字串長不能設超過9，因為以實際狀況來講不太可能要讓玩家猜這麼長的字串，故我那邊的寫法比較簡略。

總體來講這次的作業讓我學到了很多，也包括讓我稍微熟悉Java語法，還有一些特殊API，寫出來後也頗有一番感想。