需求描述：

- 本次為大富翁遊戲的第五次 checkpoint，除了前三次的基本功能外，這次 checkpoint 將更進一步實現我們印象中的大富翁遊戲，本次主要將在遊戲畫面中加上角色的圖片，並且可以讓角色移動，讓角色能跟著擲出的數字活動。需求細項如下：

(1) 本次當有任何和錢有關的操作都須即時反應在右上方的金錢欄位。

(2) 本次需連線到 140.127.220.220 的主機裡的資料庫來讀取遊戲數值。
資料欄位示意圖如下。

PLACE_NUMBER ▲ 1	LAND_PRICE	TOLLS
1	300	30
2	500	50
3	600	60

PLACE_NUMBER 為土地編號，LAND_PRICE 為土地價格，TOLLS 為過路費，一共有 16 筆資料(資料庫中目前不包含機會、命運、起點的資料)。

(3) 土地相關事件處理：

- ✓ 若玩家走到無人的地(owner 為零)，需跳出「是否購買」的視窗。
視窗的規格如下：

點 Yes 便可購買土地，點 No 則不買。若點擊 Yes 成為地主後須在土地上標示自己的編號，點 NO 則甚麼事都不會發生。如下圖：

- ✓ 若玩家踩到別的玩家的地則須繳交過路費，而地主能獲得該過路費。踩到別人的地時需跳出以下視窗：

若踩到自己的地，或踩到「機會」、「命運」則甚麼事都不會發生。

- ✓ 當剛好抵達或經過「起點」時可獲得 2000 元，且不必跳任何視窗。

(4) 要等到每位玩家的土地事件處理完了，Turn Character 輪到誰才能改變。

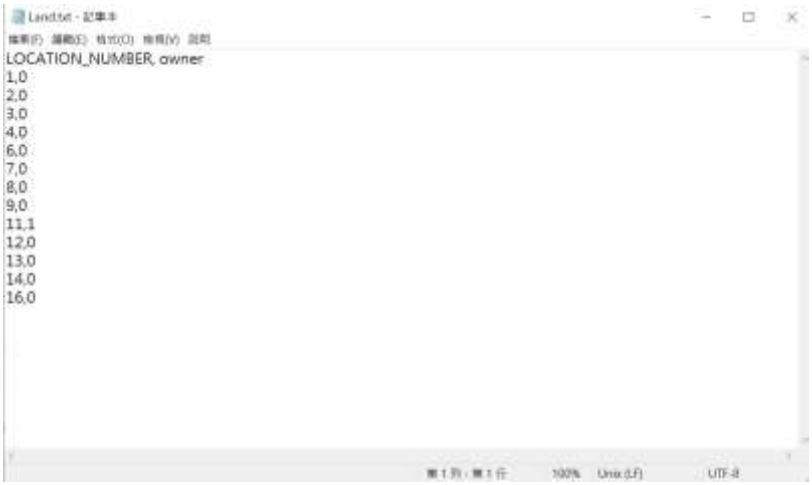
(5) 每塊土地皆需對應到一個 Land 物件(命運、起點、機會則不用)

(6) 玩家的錢可以為負，但若剩下的錢小於買地的地價則不能買地，但仍需繳交過路費。

(7) 使用 Save 時要能夠儲存，按下 Load 時要能夠讀取上次遊戲的所有數據並且把上次玩家位置和各個土地的擁有者給顯示在地圖上。

(8) 本次的開啟遊戲時需能夠讀取 Character.txt 和 Land.txt 這兩個檔案。

- ✓ Character.txt 檔案內容和上次一樣
- ✓ Land.txt 檔的內容如下：



(9) 在地圖上顯示角色圖片，每個角色要能夠根據骰子擲出來的數字前進對應的格數。角色的位置需注意以下幾點要求：

- ✓ 角色圖片要讀取附件裡的「Character_n.png」的圖片四隻可達鴨，並需依照文件中的位置擺放可達鴨。範例如下：

圖 1 角色位置擺放示意圖
(註:左上為玩家一，右上為玩家二，左下為玩家三，右下為玩家四)

- ✓ 可達鴨的擺放位置不能受到玩家人數改變而跟著改變，例如：如果目前只剩兩位玩家一號和二號，那麼可達鴨仍需擺在左上角和右上角。
- ✓ 可達鴨需要在規定時間前進到目標格子，擲出 1~3 時為兩秒到達；擲出 4~6 時為三秒到達。
- ✓ 當要轉彎的時候要以直角的方式轉彎，不能走斜的到目標格子。
- ✓ 可達鴨在走的過程中，只要保持四隻鴨子的相對位置正確即可(例如黑鴨要一職保持在白鴨的左邊)。
- ✓ 當角色走過一圈後，需將 location 歸零(最後一格為「考研」location 值為 19)。

(10) 當玩家的 status<=0 的時候需要直接跳過，表示不能輪到該玩家。

(11) 當某個角色在移動時，其他人不能進行遊戲。

(12) GUI.java 的檔案，此檔案目的為做出遊戲介面，介面的製作可透過助教資料夾裡附的圖片來做，且需依照助教規定的版面來排版。然後地圖中間要有擲骰子的按鈕和顯示遊戲資訊，資訊包括顯示擲出的數字、目前的回合數、下一個擲骰子的玩家是誰，而用點擊按鈕的方式來擲骰子。

(13) 地圖的上方需要兩個按鈕，分別是「Save」和「Load」，點擊 Save 能夠儲存目前的遊戲數據；而點擊 Load 能將上一次的數據載入遊戲裡面，從上一次的狀況繼續玩。而地圖右上角則是顯示玩家的資訊(目前為四位玩家)，包含玩家編號、玩家目前剩多少錢。

(14) 地圖的右下角要有一個「Exit」按鈕，此按鈕功能為關閉視窗結束遊戲。

(15) GUI 的介面有特定的規格：

- ✓ 不能使用網路上的 GUI 排版工具，只能用 GridLayout、FlowLayout、BorderLayout 來排版。
- ✓ 視窗大小需固定為 700*700，不能讓使用者去調整。
- ✓ GUI 讀取圖片時要相對路徑寫，不能用絕對路徑。

(16) 角色的預設值如下：

location：0
CHARACTER_NUMBER：玩家編號(1、2、3、4)
money： 2000
status：1
IMAGE_FILENAME：Character_n (n 為角色編號)。

(17) 本次附有 Checkpoint5.java 的檔案，在此檔案中，已經先附上 Load()、Save()、Random()這三個方法。Load()方法來實現讀取四位角色數據。Save()方法來實現儲存目前四位玩家的遊戲數據。

(18) 透過點擊骰子來呼叫 Random()方法，Random()用來讓 location 加上擲出的數字，代表角色前進多少格，且讓 status 減一，當 status 等於 1 代表該玩家是可以玩的狀態，而 status 歸零表示此玩家已經玩完。

OO 的考量：

- **OOA：**

土地物件：

每一塊土地應該都要視為一個物件，才能夠當使用者踩到後進行對應的處理，而每塊土地需具有編號、地主、地價、過路費等屬性

角色移動：

要讓角色能夠移動這個需求，首先要能夠存取四張圖片，四張圖片各自代表一個可操作的物件。而圖片若要能夠進行移動，可從座標著手，取得四張圖片各自的 x、y 座標，並且去計算他們各自在 x 軸移動多少，y 軸移動多少來讓圖片動起來。

訂定角色屬性：

實現大富翁之遊戲數據檔案之讀取、存檔外，還有擲骰子的功能，還有整個遊戲介面的呈現。對於數據檔案的讀取和儲存，我們可以先檢視數據檔案 Character.txt，檔案裡的 Round、Turn、location、CHARACTER_NUMBER、money、status 皆為整數，而 IMAGE_FILENAME 為圖片的檔名，所以根據這些資訊去設計一個類別，將上述的資訊轉變成該類別可擁有的屬性，例如把 Round、Turn、location、CHARACTER_NUMBER、money、status 皆宣告成整數型態的變數，而 IMAGE_FILENAME 為多個字母組合可將其形態設為 String，而由於這個類別和角色遊戲數據有關故命名 Character。

主類別：

設計另一個類別命名為 Checkpoint5，裡面要儲存、讀取、擲骰子的功能，可以分別取名為為 Save()、Load()和 Random()。而在做前兩件事情前可能需要把 txt 檔裡的資訊先存放在一個地方來方便後續的處理，因此可用陣列來存放 txt 檔的資訊，先宣告一個 String 型態的陣列 tempArray 和一個 String 型態的二維陣列 trans_array。第一個 tempArray 是先暫時存放遊戲數據，在將其整理之後再存給這個 trans_array，用意是因為 txt 檔的資料每一列便是代表一個玩家，所以第一個維度代表玩家，第二個維度代表某個遊戲屬性，如此在操作時會比較

容易知道目前操作的變數是什麼，提高易讀性。至於擲骰子的部分，可以在 Random()裡用專門生成亂數的物件來產生亂數。最後還有一個小功能是要生成一個新檔案「output.txt」，因此需要有建檔案和將資訊寫入的物件。

GUI 畫面呈現：

而關於遊戲介面的呈現，可以用另一個類別來處理，取名為 GUI。在 GUI 裡，除了上述的角色圖片要放到地圖之外，還要能夠呈現出三個按鈕，需要建立三個按鈕的物件，以及要把附件的圖片給一張張放進畫面裡，因此也可將圖片們視為一個個物件，然後右上角的資訊顯示、中間的回合顯示、下一個輪到誰的顯示也都可視為物件。由於 GUI 只負責介面的呈現所以將主程式仍放在 Checkpoint5 的 class 裡，而非在 GUI。

- **OOD：**

土地：

將從 Land.txt 和資料庫裡的資料存給每塊土地，因此每個土地需具有 owner、PLACE_NUMBER、price、tolls 這四種屬性。

資料庫：

需要有 driver、要連接的資料庫類型、資料表名稱、帳號和密碼才能夠和主機裡的資料庫建立連線。

跳出視窗：

使用 Joptionpane 來處理跳出視窗的事件，而出現的時機應該在每一個角色走完後跳出。

角色移動：

在 GUI 中將四張圖片建立為四個 ImageIcon 物件，並且幫每個圖片宣告代表各自 x、y 座標的 int 變數，並用一個新方法 paint()來處理傳入 Graphics 的參數把這四張圖片把他們畫在地圖上。而為了讓遊戲進行時，其他按鈕能夠正常運作，建一條 Thread 來分攤畫圖的工作。

因為移動過程中不能讓其他人玩，因此需將 dice 按鈕的功能暫時停掉直到該角色執行結束。

畫面呈現：

GUI 的建構式中需要的物件和屬性：需要建立多個 JButton 的物件來處理畫面中的按鈕，建立多個 JPanel 的物件來排版排出指定的遊戲畫面，從最上排的角色訊息、中央的地圖、和最下面的 Exit 按鈕，而中間的地圖需用多個 ImageIcon 的物件來讀取附件的圖片來呈現地圖，建立多個 JLabel 的物件來處理右上角和中間的顯示資訊的部分。

角色屬性：

在 OOA 提到有 Character 類別，接著將詳細定義該類別，此類別內具有 location、CHARACTER_NUMBER、money、status、IMAGE_FILENAME 這五個屬性，前四個宣告為 int，IMAGE_FILENAME 宣告為 String 型態，而 Character 只有一個不用傳入任何參數的建構式，就先將各個數值初始化把 int 型態的變數先設為零，而 String 型態的先設為空字串。

主類別：

再來定義 Checkpoint 類別，該類別有 String 型態的陣列 tempArray、String 型態的二維陣列 trans_array、和 int 型態的變數 k，k 是用在後續迴圈中判斷做幾次的依據，而宣告動態陣列 myList 來應對當玩家不只二人的時候，即使玩家資料變多也能存入 myList。

因為在其他的功能會需要呼叫到這些變數，因此將這些屬性皆宣告成 public static。

Load()方法所需要的物件和屬性：讀取兩個 txt 檔的方法大致上是一樣的。需要 FileReader 和 BufferedReader 的物件來讀取檔案，而為了存取檔案中的數據，宣告兩個 String 型態的變數 line 和 tempString 相繼來接讀取到的值。

Save()方法所需要的物件和屬性：需要二個 FileWriter 的物件用於寫入檔案，一個用於

Character.txt，另一個用於 Land.txt，而 save 裡不需要特別再建立其他新的變數。
Random()方法所需要的物件：需利用 trans_array 和 act 作為判斷條件，在特定的條件下，對特定的物件進行操作，例如 JLabel 的 round2 和 role2、全域變數 Numround 等。
在主程式中需要的物件和屬性：k 是一個代表玩家數量的 int 變數，Numoround 是代表回合數的 int 變數，randomNum 是用來存放亂數的 int 變數，而 int 型態的 act 是用來判讀輪到哪個角色的計數器。然後建立 trans_array，把它建成一個陣列大小為[k][5]的二維陣列，建立一個 Character 類別的物件陣列大小為 k，建立一個 File 類別的物件 picture 和 BufferedImage 的物件 bufferImage 來做為讀圖片之用途。

- **OOP：**

主類別：

- ✓ 在 Load()中先處理 Character，要做的是先傳入 String 型態的 filename 參數到方法裡面，這些數據先以 FileReader 和 BufferedReader 讀檔後，透過 tempArray 在經由 myList 的處理後存放在 trans_array 的陣列裡面。而另一個 Land 檔案則需要透過另一個動態陣列 myList2 來處理，而處理方法和上面相同。
- ✓ 在 Save()中要做的是先傳入 String 型態的 filename 參數到方法裡面，透過 FileWriter 來把資料重新寫入 Character.txt。
- ✓ 在 Random()中不需要傳入任何參數，但須透過全域變數 trans_array[][1](也就是 status)和 act 當下的數值來判斷哪號玩家且他的值是否為零來進行一連串的判斷，最基本的當條件成立讓該玩家的 location 加上亂數 1~6，和讓 status 減 1 的運算。
- ✓ 和主機的資料庫連線成功後，將讀取到的資料各自建成一個動態陣列，因此有三個陣列能夠，由於三種資料都很高頻率在後面的程式中出現，如此才能方便在後續的操作。
- ✓ 在主程式中應先呼叫 Load()方法作為開始，因為 Load 負責把 txt 檔的資料讀出來，要先有檔案的數據才能夠做後續的動作。然後每個玩家可以視為物件陣列 ccc[]裡的成員，把 trans_array 的值當作參數存進每一個對應的物件陣列 ccc[k]的建構式裡，以完成玩家數據的建立。接著建立 GUI 類別的物件，讓主程式能夠把遊戲畫面叫出來。

遊戲畫面：

- ✓ 而 GUI 的部分：先讓 GUI 類別去繼承 JFrame 類別讓 GUI 直接使用 JFrame 中的方法(例如 setVisible())。接著設計 GUI 的建構式，為了實現規定的遊戲畫面，可以先將整個遊戲畫面分成三個大 panel(上排、中間地圖、下排)，上牌可在拆成左右兩個小 panel 來分別放按鈕和玩家資訊表；中間外圈的地圖可拆成上下左右各一個 panel，這四個 panel 裡有存放多個 Jlabel，而這些 JLabel 裡則是存放著地圖的圖片，再來是地圖的中間也是設置一個大 panel 並將此 panel 分成三個小 panel，第一小塊處理標題圖片，第二小塊處理 dice 按鈕和顯示數字格，第三小塊處理回合數和下一位輪到誰的顯示。下排則是僅有一個按鈕而已。而當按鈕被按下時，透過 actionPerformed 的方法來產生相關的對應程序，傳入的參數為發生的事件 ActionEvent，當按下 Load 按鈕需呼叫 Load()方法，按下 Save 按鈕時呼叫 Save()方法，而按下 Dice 按鈕時則呼叫 Random()方法。
- ✓ 在 GUI 建構式的尾端加上 goback 的方法讓遊戲一打開角色便能在上次結束的位置出現。
- ✓ 而按下 Load 後要能夠讓角色們回到上一次的位置(根據最新一次存檔的紀錄)，透過設計一個 goback 的方法來實現，概念是利用 run()裡判斷轉彎直走的觀念，判斷完直走或轉彎多少步後，讓程式只 repaint 一次，即可將角色歸位。
- ✓ 先讓 GUI class 去實作 Runnable，透過 Duckstart()方法先建立出新的 thread 處理圖片相關的操作。而圖片需先在 paint()方法裡透過 getImage()方法才能將它們放在地圖上，並且將四張圖片的座標參數傳入 getImage()裡，讓角色根據座標去移動，而移動的

方式透過 Thread 的 run() 方法內來判斷要轉彎或直走的時機。因為角色要在擲出骰子後移動，所以 Random() 中呼叫此方法。而要讓骰子自動關掉和開啟則是用 change_dice 來切換。

- ✓ 擲完骰子，然後在 run() 裡判斷走路的方式後，走到某塊地後根據土地的狀況判斷要跳出哪種的土地視窗或是不跳視窗。而跳出視窗的方式可使用 JOptionpane 中的 showMessageDialog，此種 optionpane 會自動預設 yes 和 no 的按鈕就不必自己在手動設按鈕。

功能/邏輯說明：

- **說明連接資料庫的部分：**

  一開始先載入 driver，並且設定好要用的資料庫類型、帳號、密碼、資料表名稱、主機位址等資訊。

```
324     public static void DB() throws IOException
325     {
326          String driver ="com.mysql.cj.jdbc.Driver";
327          String protocol = "jdbc:mysql:";
328          try
329          {
330              Class.forName(driver).newInstance();
331              System.out.println("Loaded the embedded driver.");
332          }
333          catch (Exception err)
334           {
335                System.err.println("Unable to load the embedded driver.");
336                err.printStackTrace(System.err);
337                System.exit(0);
338
339           }
```

  連接資料庫並且開始下 query 指令，建立三個 statement 和三個 query，將三種土地資料抓出來。

```
340          String url = "//140.127.220.220/";
341          String dbName = "CHECKPOINT";
342          Connection conn = null;
343          String username = "checkpoint";
344          String password = "ckppwd";
345
346          try
347          {
348          System.out.println("Connecting to and creating the database...");
349          conn = DriverManager.getConnection(protocol + url + dbName +"?serverTimezone=UTC", username, password);
350          Statement s = conn.createStatement();
351          Statement s2 = conn.createStatement();
352          Statement s3 = conn.createStatement();
353          String qry1 = "select LAND.PLACE_NUMBER from LAND";
354          String qry2 = "select LAND.land_price from LAND";
355          String qry3  = "select LAND.tolls from LAND";
356          ResultSet rs1 = s.executeQuery(qry1);
```

  把 query 出來的三種資料利用 while 迴圈和 next() 各自存入動態陣列裡。存完後關閉資料庫。

```
357         while(rs1.next())
358         {
359         int place=rs1.getInt("PLACE_NUMBER");
360         place_number.add(place);
361         System.out.println(place+"資料庫place");
362         }
363         System.out.println(place_number);
364         ResultSet rs2 = s2.executeQuery(qry2);
365         while(rs2.next())
366         {
367         int landprice=rs2.getInt("LAND_PRICE");
368         land_price.add(landprice);
369         System.out.println(landprice+"資料庫price");
370         }
371         ResultSet rs3 = s3.executeQuery(qry3);
372         while(rs3.next())
373         {
374         int tolls=rs3.getInt("TOLLS");
375         tolls_array.add(tolls);
376         System.out.println(tolls+"資料庫toll");
377         }
378
379         conn.close();
380         }catch (SQLException err)
381         {
382               System.err.println("SQL error.");
383               err.printStackTrace(System.err);
384               System.exit(0);
385          }
386
387
388
389     }
```

- **跳出土地視窗：**

處理跳出視窗的方法名為 jumpwindow，此方法需傳入 role_num 變數代表目前輪到哪個玩家。此方法的邏輯為將玩家的骰子點數累加，用累加點數(變數 duck_dice[])判斷落在哪個土地區間(0~5、5~10、10~15、15~19)，接著再判斷該土地地主是誰，如果是別人就繳過路費，如果是無人地就跳出是否購買訊息框。並且在 run()方法中呼叫此方法。下面以其中一種區間的程式碼做說明，其他區間的以此類推。

```
716     public void jumpwindow(int role_num)
717     {
718
719         duck_dice[role_num]=A1063329_Checkpoint5.randomNum+duck_dice[role_num];
720         //System.out.println(duck_dice[role_num]+"duckdice");
721         //int tmp=duck_dice[role_num]-4;
722         if(duck_dice[role_num]>=20)
723         {
724
725                 A1063329_Checkpoint5.trans_array[role_num][2]=Integer.toString(Integer.parseInt(A1063329_Checkpoint
726                 money[role_num].setText(A1063329_Checkpoint5.trans_array[role_num][2]);
727                 duck_dice[role_num]=duck_dice[role_num]-20;
728         }   //walk for one round
729
730         if( duck_dice[role_num]>0 && duck_dice[role_num]<5)
731         {   //duck_dice[role_num]==A1063329_Checkpoint5.place_number.get(duck_dice[role_num]-1)
732             System.out.println("jumpddd");
```

判斷 owner 是不是自己，不是就扣錢並跳出視窗；是的話不做任何動作。而如果是踩到無人地則跳出詢問購買視窗，如果按下 Yes 按鈕(YES_OPTION)的話玩家扣錢並且在該土地上 setText()自己的角色編號，然後在把 owner 設成自己。

- ✓ Joptionpane 利用 String.format 把要顯示的話和變數設定在視窗上。
- ✓ 扣錢的方式：trans_array[][2]代表玩家的錢減掉 land_price。
- ✓ 透過 whole_land[]的索引值來確認要在哪塊土地上 setText。
- ✓ trans_array2[][2]表示 owner。

```
733     System.out.println("duck_dice[role_num]-1="+(duck_dice[role_num]-1));
734     int tempInt=Integer.parseInt(A1063329_Checkpoint5.trans_array[role_num][2]);
735
736     if(A1063329_Checkpoint5.trans_array2[duck_dice[role_num]-1][1]!=(role_num+1) && A1063329_Checkpoint5.trans_arrc
737     {
738         JOptionPane.showMessageDialog(A1063329_GUI.this,String.format("this land belongs to Character %d please pay
739                 A1063329_Checkpoint5.tolls_array.get(duck_dice[role_num]-1)),
740                 "PAY THE TOLLS!", JOptionPane.INFORMATION_MESSAGE);
741         A1063329_Checkpoint5.trans_array[role_num][2]=Integer.toString(tempInt-A1063329_Checkpoint5.tolls_array.get
742         int tmpOwner=A1063329_Checkpoint5.trans_array2[duck_dice[role_num]-1][1]-1;
743         A1063329_Checkpoint5.trans_array[tmpOwner][2]=Integer.toString(Integer.parseInt(A1063329_Checkpoint5.trans_
744         money[tmpOwner].setText(A1063329_Checkpoint5.trans_array[role_num][2]);
745     //buyland
746     }else if(A1063329_Checkpoint5.trans_array2[duck_dice[role_num]-1][1]==0 && tempInt>0 && tempInt>A1063329_Checkp
747     {
748     result4=JOptionPane.showConfirmDialog(A1063329_GUI.this,String.format("Do you want to buy?%d", A1063329_Checkpc
749             JOptionPane.YES_NO_OPTION,mType);
```

```
750      if (result4==JOptionPane.YES_OPTION)
751      {
752         A1063329_Checkpoint5.trans_array[role_num][2]=Integer.toString(tempInt-A1063329_Checkpoint5.land_price.get(
753         whole_land[A1063329_Checkpoint5.place_number.get(duck_dice[role_num]-1)].setHorizontalTextPosition(0);
754         whole_land[A1063329_Checkpoint5.place_number.get(duck_dice[role_num]-1)].setText(A1063329_Checkpoint5.trans
755         A1063329_Checkpoint5.trans_array2[duck_dice[role_num]-1][1]=role_num+1;
756      }
757     }
```

- **說明 GUI 角色移動的部分：**

宣告 ImageIcon duck[]的陣列存四張圖片。

```
285     duck[0] = new ImageIcon("src//checkpoint4//Character_1.png");
286     duck[1] = new ImageIcon("src//checkpoint4//Character_2.png");
287     duck[2] = new ImageIcon("src//checkpoint4//Character_3.png");
288     duck[3] = new ImageIcon("src//checkpoint4//Character_4.png");
```

透過 paint()來畫圖，方法內先 super.paint 繼承 Graphic 類別裡的方法，接著傳入圖片和其 x、y 座標的參數到 drawImage()裡來把圖片畫在地圖上。

```
296     public void paint(Graphics g) {
297         super.paint(g);
298         g.drawImage(duck[0].getImage(), duckstart_x1, duckstart_y1, null);
299         g.drawImage(duck[1].getImage(), duckstart_x2, duckstart_y2, null);
300         g.drawImage(duck[2].getImage(), duckstart_x3, duckstart_y3, null);
301         g.drawImage(duck[3].getImage(), duckstart_x4, duckstart_y4, null);
302
303     }
```

在 Duckstart()中創建 Thread t1，並且啟動它 start()。並且在 Random()中呼叫此方法。

```
588         public void Duckstart()
589         {
590
591             Thread t1=new Thread(this);
592             t1.start();
593
594         }
```

接著 override 執行序的 run()，這個部分程式的過程是先判斷輪到哪個角色能動(role_num 的值)，然後再根據擲出的數字乘以 90(一張圖片的寬度)，判斷 duckstart_x 或 duckstart_y 要走多少步，而為了要讓圖片有像動畫動的感覺，因此每次的變動量都是 1，處理角色前進的方式。前進的方式可分為兩種狀況直線前進和轉彎，直線前進的處理方式是將 y 軸的變量固定，只改變 x 軸的值；而轉彎的部分有四個轉角，因此要先算出四個轉角的座標(可根據使用者的設計而有不一樣，沒有強制規定特定的座標)再根據該轉角座標決定增或減 x、y 軸的變動量。判斷完怎麼動之後呼叫 repaint 重畫，透過 Thread.sleep()

睡的時間來控制 repaint 的速度藉此需求中規定移動時間 2 秒和 3 秒。而 change_dice() 的方法是要讓 dice 按鈕在某角色移動時，其他角色不能去按 dice，直到該角色動完再把按鈕恢復成可以按的狀態。以下為角色 1 號的判斷程式碼，其他人以此類推。

```
309     public void run() {
310
311 //---------------------------------DUCKx1
312         no_press_dice++;
313         change_dice();
314         if(role_num==0)
315         {
316
317             move_counter=0;
318         while(move_counter<A1063329_Checkpoint4.randomNum*90)
319         {
320             if(duckstart_x1>=90 && duckstart_y1==600)
321             {
322                 duckstart_x1--;
323                 move_counter++;
324                if(duckstart_x1<90)
325                {
326                    move_counter=move_counter-55;
327                }
328            }
329            if(duckstart_x1==89 && duckstart_y1>=120)
330            {
331                duckstart_y1-- ;
332                move_counter++;
333                if(duckstart_y1<120)
334                {
335                    move_counter=move_counter-55;
336                }
337            }
338            if(duckstart_x1<=550 && duckstart_y1==119)
339             {
340                 duckstart_x1++;
341                 move_counter++;
342                 if(duckstart_x1>550)
343                 {
344                     move_counter=move_counter-55;
345                 }
346             }
347             if(duckstart_x1==551 && duckstart_y1<=600)
348             {
349                 duckstart_y1++;
350                 move_counter++;
351                 if(duckstart_y1>600)
352                 {
353                     move counter=move counter-55;
```

```
353                 move_counter=move_counter-55;
354             }
355         }
356
357         if(A1063329_Checkpoint4.randomNum==1
358                 ||A1063329_Checkpoint4.randomNum==2||A1063329_Checkpoint4.randomNum==3)
359             {
360                 this.repaint();
361                 try {Thread.sleep((1000*2)/(110*A1063329_Checkpoint4.randomNum));
362                 }catch(InterruptedException e)
363                 {
364                     e.printStackTrace();
365                 }
366             }else
367             {
368                 this.repaint();
369                  try {Thread.sleep(1000*3/(110*A1063329_Checkpoint4.randomNum));
370                  }catch(InterruptedException e)
371                  {
372                      e.printStackTrace();
373                  }
374              }
375         }
376         change_dice();
377         }
```

- **說明 GUI 的排版的部分：**

建構式內一開始先用 super()呼叫父類別的建構式，然後設定視窗大小，設定關閉時的方式為 EXIT_ON_CLOSE，設定 GUI 的排版方式為 BorderLayout。先宣告 Container 的物件名為 contentpane 然後用 getContentPane()取得 JFrame 的內容面板，接著設定兩個 JPanel，分別為 blank1、blank2，並且設定這兩塊的大小 setPreferredSize(new Dimension(25,280))，然後將這兩塊 panel 加到 contentpane 裡，設置兩塊空白的 panel 來讓調整遊戲介面，使中間的地圖長相能夠較符合規定。

```
40-     public  A1063329_GUI() {
41         super();
42         setVisible(true);
43         setResizable(false);
44         setSize(WIDTH,HEIGHT);
45         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
46         setLayout(new BorderLayout());
47
48         Container contentPane = this.getContentPane();
49         JPanel blank1 =new JPanel();
50         blank1.setPreferredSize(new Dimension(25,280));
51         JPanel blank2 =new JPanel();
52         blank2.setPreferredSize(new Dimension(25,280));
53         contentPane.add(blank1,BorderLayout.WEST);
54         contentPane.add(blank2,BorderLayout.EAST);
```

先從地圖上面的部份開始。設 JPanel panel1，在這塊 panel1 上左右各放一塊小 panel，分別 smallpanel1 和 smallpanel2。將 save 和 load 的按鈕加在 smallpanel1 上，而右邊的 smallpanel2 顯示角色數據，用兩個 for 來處理顯示的資料，第一個 for 將每個 CHRARCTER_NUMBER 設在 label 上並加到 smallpanel2 裡，第二個 for 將每個角色的 money 設在一個個 label。

```
40-    public  A1063329_GUI() {
41         super();
42         setVisible(true);
43         setResizable(false);
44         setSize(WIDTH,HEIGHT);
45         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
46         setLayout(new BorderLayout());
47
48         Container contentPane = this.getContentPane();
49         JPanel blank1 =new JPanel();
50         blank1.setPreferredSize(new Dimension(25,280));
51         JPanel blank2 =new JPanel();
52         blank2.setPreferredSize(new Dimension(25,280));
53         contentPane.add(blank1,BorderLayout.WEST);
54         contentPane.add(blank2,BorderLayout.EAST);
55 //         JPanel panel1=new JPanel();
56         panel1.setLayout(new GridLayout(1,2));
57         JPanel smallpanel=new JPanel();
58         smallpanel.setLayout(new FlowLayout());
59         JButton bsave=new JButton("save");
60
61         bsave.addActionListener(this);
62         smallpanel.add(bsave);
63
64
65         JButton bload=new JButton("load");
66 //         JLabel lload=new JLabel();
67         bload.addActionListener(this);
68         smallpanel.add(bload);
69         panel1.add(smallpanel);
```

設一個處理中央地圖的 panel 叫 bigmap，用 BorderLayout 排版。首先處理地圖外圈上排的部分：

設 JPanel map，排版方式為 GridLayout。宣告好幾個 ImageIcon 的物件來讀取附件裡的地圖格子圖片，並且將每個圖片加到 label 裡(一張圖放一個 label)，再將這些 label 加到 map 裡，在把 map 加到 bigmap 的上方。

```
93         JPanel bigmap=new JPanel();
94         bigmap.setLayout(new BorderLayout(0,0));
95         JPanel map=new JPanel();
96         map.setLayout(new FlowLayout());
97         //map.setLayout(new GridLayout(1,6,0,0));
98         ImageIcon png10 = new ImageIcon("src\\checkpoint3\\10.png");
99         JLabel chance=new JLabel(png10);
100        ImageIcon png11 = new ImageIcon("src\\checkpoint3\\11.png");
101        JLabel sad=new JLabel(png11);
102        ImageIcon png12 = new ImageIcon("src\\checkpoint3\\12.png");
103        JLabel oop2=new JLabel(png12);
104        ImageIcon png13 = new ImageIcon("src\\checkpoint3\\13.png");
105        JLabel os=new JLabel(png13);
106        ImageIcon png14 = new ImageIcon("src\\checkpoint3\\14.png");
107        JLabel p1=new JLabel(png14);
108        ImageIcon png15 = new ImageIcon("src\\checkpoint3\\5.png");
109        JLabel destiny15=new JLabel(png15);
```

```
110         map.add(chance);
111         map.add(sad);
112         map.add(oop2);
113         map.add(os);
114         map.add(p1);
115         map.add(destiny15);
116         bigmap.add(map,BorderLayout.NORTH);
```

處理地圖外圈左排的部分：

設 panel 名為 map2，排版方式為 GridLayout。先宣告多個 ImageIcon 的物件來讀取圖片，並將圖片加到 label 裡(一張圖放一個 label)，再將這些 label 加到 map2 裡，在把 map 加到 bigmap 的左方。

```
119         JPanel map2=new JPanel();
120         //map22.setLayout(new FlowLayout());
121         map2.setLayout(new GridLayout(4,1,0,0));
122         ImageIcon png9 = new ImageIcon("src\\checkpoint3\\9.png");
123         JLabel database=new JLabel(png9);
124
125         ImageIcon png8 = new ImageIcon("src\\checkpoint3\\8.png");
126         JLabel php=new JLabel(png8);
127         ImageIcon png7 = new ImageIcon("src\\checkpoint3\\7.png");
128         JLabel internet=new JLabel(png7);
129         ImageIcon png6 = new ImageIcon("src\\checkpoint3\\6.png");
130         JLabel market=new JLabel(png6);
131
132
133         map2.add(database);
134         map2.add(php);
135         map2.add(internet);
136         map2.add(market);
137         bigmap.add(map2,BorderLayout.WEST);
```

處理地圖外圈右排的部分：

設 panel 名為 map3，排版方式為 GridLayout。先宣告多個 ImageIcon 的物件來讀取圖片，並將圖片加到 label 裡(一張圖放一個 label)，再將這些 label 加到 map3 裡，在把 map3 加到 bigmap 的右方。

```
139         //---------------------map3
140         JPanel map3=new JPanel();
141         map3.setLayout(new GridLayout(4,1,0,0));
142
143         ImageIcon png16 = new ImageIcon("src\\checkpoint3\\16.png");
144         JLabel ethic=new JLabel(png16);
145
146         ImageIcon png17 = new ImageIcon("src\\checkpoint3\\17.png");
147         JLabel p2=new JLabel(png17);
148         ImageIcon png18 = new ImageIcon("src\\checkpoint3\\18.png");
149         JLabel inter=new JLabel(png18);
150         ImageIcon png19 = new ImageIcon("src\\checkpoint3\\19.png");
151         JLabel master=new JLabel(png19);
152
153
154         map3.add(ethic);
155         map3.add(p2);
156         map3.add(inter);
157         map3.add(master);
158         bigmap.add(map3,BorderLayout.EAST);
```

處理地圖外圈下排的部分：

設 panel 名為 map4，排版方式為 GridLayout。先宣告多個 ImageIcon 的物件來讀取圖片，並將圖片加到 label 裡(一張圖放一個 label)，再將這些 label 加到 map4 裡，在把 map4 加到 bigmap 的下方。

```
162         JPanel map4=new JPanel();
163         //map4.setLayout(new GridLayout(1,6,0,0));
164         map4.setLayout(new FlowLayout());
165         ImageIcon png0 = new ImageIcon("src\\checkpoint3\\0.png");
166         JLabel start=new JLabel(png0);
167         ImageIcon png4 = new ImageIcon("src\\checkpoint3\\4.png");
168         JLabel oop1=new JLabel(png4);
169         ImageIcon png3 = new ImageIcon("src\\checkpoint3\\3.png");
170         JLabel math=new JLabel(png3);
171         ImageIcon png2 = new ImageIcon("src\\checkpoint3\\2.png");
172         JLabel c=new JLabel(png2);
173         ImageIcon png1 = new ImageIcon("src\\checkpoint3\\1.png");
174         JLabel management=new JLabel(png1);
175         ImageIcon png5 = new ImageIcon("src\\checkpoint3\\5.png");
176         JLabel destiny5=new JLabel(png5);
177         map4.add(destiny5);
178         map4.add(oop1);
179         map4.add(math);
180         map4.add(c);
181         map4.add(management);
182
183         map4.add(start);
184         bigmap.add(map4,BorderLayout.SOUTH);
```

處理地圖中央的部分：

先設一個大 panel 叫做 centermap，再設三個小 panel，分別是 center1、center2、center3 來處理中央的排版，排版方式為 GridLayout。

center1 的部分，將圖片 title.png 放在一個 label 裡，將該 label 加入到 center1 的左邊。

```
187         JPanel centermap=new JPanel();
188         //centermap.setLayout(new GridLayout(3,1));
189         centermap.setLayout(new BorderLayout());
190         ImageIcon title = new ImageIcon("src\\checkpoint3\\title.png");
191         JPanel center1=new JPanel();
192         center1.setLayout(new BorderLayout());
193         JLabel jtitle=new JLabel(title,JLabel.LEFT);
194         center1.add(jtitle,BorderLayout.WEST);
```

center2 的部分，排版方式為 GridLayout。將 Dice.png 圖片放入 jdice 這個 JButton 裡，並且將 display_dicenum.png 放到另一個 label 裡，最後將兩個 label 加入 center2 裡。

```
196         //-------------------------dice
197         JPanel center2=new JPanel();
198         center2.setLayout(new GridLayout(1,2));
199         ImageIcon dice = new ImageIcon("src\\checkpoint3\\Dice.png");
200
201         JButton jdice=new JButton();
202         jdice.setOpaque(false);
203         jdice.setBorderPainted(false);
204         jdice.setContentAreaFilled(false);
205         jdice.setActionCommand("dice");
206         jdice.setIcon(dice);
207         jdice.addActionListener(this);
208
209
210         center2.add(jdice);
211         center2.add(Jdisplay);
```

先找出進入遊戲後輪到誰先玩，再讓 role2 印出來。宣告 int count=0，接著用 for 迴圈裡頭第一個 if 判斷哪一位玩家 status 不是零，如果成立的話，就讓 role2 印出該玩家編號，然後 break 跳出迴圈；第二個 if 判斷 count 是否等於 4，就讓 role2 印出第一位玩家的編號後 break 跳出迴圈。

```
227     int count=0;
228         for(int i=0;i<A1063329_Checkpoint3.myList.size()/5;i++) {
229             count++;
230             if(Integer.parseInt(A1063329_Checkpoint3.trans_array[i][3])!=0) {
231                 role2.setText("Turn Character "+A1063329_Checkpoint3.trans_array[i][1]);
232                 break;
233             }
234             if(count==4) {
235                 role2.setText("Turn Character "+A1063329_Checkpoint3.trans_array[0][1]);
236                 break;
237             }
238
239         }
```

center3 的部分排版方式為 GridLayout，將兩個 label(round2、role2)，這兩個 label 需調整字體大小來符合助教附件的範例圖，最後分別放入新的 panel 名為 minipanel2，再將 panel 加到 center2 裡。

```
259         round2.setText("Round "+A1063329_Checkpoint3.Numround);
260 //      role2.setText("Turn Character ");
261         round2.setHorizontalAlignment(SwingConstants.LEFT);
262         role2.setHorizontalAlignment(SwingConstants.LEFT);
263         round2.setFont(new Font("",Font.BOLD, 22));
264         role2.setFont(new Font("",Font.BOLD, 22));
265
266         JPanel  minipanel=new JPanel();
267         JPanel  minipanel2=new JPanel();
268         JPanel  minipanel3=new JPanel();
269         minipanel2.setLayout(new GridLayout(3,1,8,4));
270         minipanel2.add(round2);
271         minipanel2.add(role2);
272         minipanel2.add(minipanel3);
273
274         center3.setLayout(new GridLayout(1,2));
275         center3.add(minipanel);
276         center3.add(minipanel2);
```

最後將 center1(放在上方)、center2(放在中間)、center3(放到北邊)加入到 centermap 的中間裡。

```
280         centermap.add(center1,BorderLayout.NORTH);
281         centermap.add(center2,BorderLayout.CENTER);
282         centermap.add(center3,BorderLayout.SOUTH);
283         bigmap.add(centermap,BorderLayout.CENTER);
284
285
286         //-------------------------
287
288         add(bigmap,BorderLayout.CENTER);
```

處理地圖下方的部分:下方設一個 panel 叫 panel3,排版方式為 BoderLayout。在 panel 的右邊加上一個結束程式按鈕名稱為 exit,按鈕加上 addActionListener 來跑按下後要做的事。

```
290         //-----------------------------exit
291         JPanel panel3=new JPanel();
292         panel3.setLayout(new BorderLayout());
293         JButton bexit=new JButton("exit");
294         bexit.addActionListener(this);
295         panel3.add(bexit,BorderLayout.EAST);
296         add(panel3,BorderLayout.SOUTH);
297
298     }
```

事件處理的部分：

實作 actionPerformed 的方法來處理按下按鈕的事件。當按下按鈕為「dice」，呼叫 Random 方法，將 randomNum(骰子點數)設給 str，將 str 的內容放到 Jdisplay 這個 label 上顯示，並且 setFont 調整字體大小，和 setHorizontalTextPosition(0)將顯示數字置中。

```
605     public void actionPerformed(ActionEvent e) {
606         // TODO Auto-generated method stub
607         String buttonString=e.getActionCommand();
608
609         if(buttonString.equals("dice"))
610         {
611         A1063329_Checkpoint4.Random();
612         Duckstart();
613
614         System.out.println("dice : "+A1063329_Checkpoint4.randomNum);
615         str= Integer.toString(A1063329_Checkpoint4.randomNum);
616         Jdisplay.setHorizontalTextPosition(0);
617         Jdisplay.setFont(new Font("",Font.BOLD, 28));
618         Jdisplay.setText(str);
619
620     }
```

當按下按鈕為「save」，呼叫 Save 方法，並加上 try&catch 在發生例外時，呼叫 printStackTrace()，檢視程式碼錯在哪一行。

```
359         else if (buttonString.equals("save"))
360         {
361             try {
362                 A1063329_Checkpoint3.Save("src\\checkpoint3\\Character.txt");
363             } catch (IOException e1) {
364                 // TODO Auto-generated catch block
365                 e1.printStackTrace();
366             }
367         }
```

當按下按鈕為「load」，先將畫面隱藏 setVisible(false)，然後才呼叫 Load 方法來讀 Character.txt 裡的資料，讀到資料後先接著用一個 for 迴圈將玩家編號寫在右上角，再用另一個 for 迴圈把玩家的錢給印出來在玩家編號的下方，如此一來按下 load 按鈕時便能更新資料。最後在把 setVisible()的參數設為 true，將畫面顯現出來，一開始先穎藏最後在顯現視窗的原因是希望當點下按鈕後有程式在跑，跑完後跳出新畫面的感覺。

```
368         else if (buttonString.equals("load"))
369         {
370             try {
371                 setVisible(false);
372                 A1063329_Checkpoint3.Load("src\\checkpoint3\\Character.txt");
373                 for(int i=0;i<A1063329_Checkpoint3.myList.size()/5;i++) {
374 
375                     ch[i].setText("Character"+A1063329_Checkpoint3.trans_array[i][1]);
376 
377                     }
378                     for(int i=0;i<A1063329_Checkpoint3.myList.size()/5;i++) {
379 
380                         money[i].setText(A1063329_Checkpoint3.trans_array[i][2]);
381 
382                     }
383 //                  panel1.add(smallpanel2);
384                     setVisible(true);
385             } catch (IOException e1) {
386                 // TODO Auto-generated catch block
387                 e1.printStackTrace();
388             }
389 
```

用 for 迴圈判斷到有人走完一圈(也就是 location>=19)，則將 location 減掉 19 後並多出來的部分繼續走。

```
137         for(int a=0;a<myList.size()/5;a++)
138         {
139             if(Integer.parseInt(trans_array[a][0])>=19)
140             {
141 
142                 trans_array[a][0]=Integer.toString(Integer.parseInt(trans_array[a][0])-19);
143             }
144         }
```

當按下監聽事件按鈕為「exit」，將程式結束，用 System.exit(0)把程式給關起來。

```
391         else if(buttonString.equals("exit"))
392         {
393             System.exit(0);
394         }
395
396 }
397
398 }
```

- **說明 Checkpoint5.java 的部分：**

主程式：

讓 main throws IOException 當發生異常時拋出錯誤訊息。而其他說明如下：

(1) 先呼叫 Load()，並傳入 Character.txt 的位置來讀取檔案。

(2) 建立物件陣列，ccc=new Character[k]。由於這次的 Character.java 檔案中附有建構式，因此用 for 迴圈來將參數 傳入物件陣列裡。

(3) 建立 A1063329_GUI 類別的物件名為 gui。

(4) 將 gui 顯示出來，用 gui.setVisible(true)。

```
42  public static void main(String[] args) throws IOException{
43
44      //// TODO: Announce your GUI object to make the GUI ////
45      //// TODO: For this work if you make every Character object to Random(), then you should
46      //// Hint: How to put the image
47      //// Hint: first you should announce a ImageIcon e.g. ImageIcon image = new ImageIcon(Ima
48      //// Hint: then put it into layout object what you want to display on. e.g. JButton btn =
49      Load("Character.txt");
50      k=myList.size()/5;
51      ccc=new Character[k];
52      for(int i=0;i<k;i++) {
53          ccc[i]=new Character(Integer.parseInt(trans_array[i][0])
54                  ,Integer.parseInt(trans_array[i][1]),Integer.parseInt(trans_array[i][2]),
55                  Integer.parseInt(trans_array[i][3]),trans_array[i][4]);
56
57      }
58
59      A1063329_GUI gui=new A1063329_GUI();
60      gui.setVisible(true);
61  }
```

Load 方法：

以一個 static method 叫做 Load()來做為讀取角色資料的方法，並且需傳入型態為 String 的檔案名稱(filename)，要傳入兩個，讓 Load 知道要讀取的對象。並且讓 Save throws IOException 當發生異常時拋出錯誤訊息。而方法內部說明如下：

(1) 在 Load()裡面使用 FileReader fr 和 BufferedReader br 來讀檔案。

(2) 接著宣告一個字串陣列 tempArray=new String[5]和動態陣列 myList。

(3) 因為第一行為 Turn 和 Round，所以用 toString()的方法把各自的數值讀出來。

(4) 接著用 while 迴圈，判斷條件為是否還有下一行可讀 br.readLine()!=null。，因為 Character.txt 裡的數據是以逗點隔開，所以迴圈內部先用 split(",")將逗點處理起來後，將各個數值分開後丟入 tempArray 裡。接著再用一個 for 迴圈將 tempArray 的值一個個加到 myList 裡面，如此 myList 裡就會擁有每個角色的數值。

(5) 接著將 trans_array 設成 String 型態的二維陣列，利用 for 迴圈並且搭配用 myList.get()將 myList 裡的值存出來給 trans_array。存成 trans_array 的目的是為了讓自己在使用時方便看出來目前是處理哪個玩家的哪個屬性(例如 trans_array[1][0]就是第二位玩家的 location 屬性)。

```
65    public static void Load(String filename, String filename2) throws IOException {
66        //// TODO: You should load the variables from the files. ////
67            FileReader fr = new FileReader(filename);
68            BufferedReader br = new BufferedReader(fr);
69            String line,tempstring;
70            tempArray= new String[5];
71            myList = new ArrayList<String>();
72            int i=0;
73            String fline=br.readLine();
74             subround= fline.substring(6,7);
75             subturn= fline.substring(13,14);
76           // System.out.println(subturn);
77            while((line = br.readLine())!=null)
78             while((line = br.readLine())!=null)
79             {
80                  tempstring = line;
81
82                tempArray = tempstring.split(",");
83
84                 //line=br.readLine();
85
86               for(i=0;i< tempArray.length;i++)
87                   {
88                       myList.add(tempArray[i]);
89                   }
90
91               }
93            br.close();
94
95            k = myList.size()/5;
96               int count=0;
97               trans_array = new String[k][5];
98              // Character[] c =new Character[k];
99               for(int x=0;x<myList.size()/5;x++)
00               {
01               //c[x]=new Character();
02                for(int y=0;y<5;y++)
03                   {
04                       trans_array[x][y]=(String)myList.get(count);
05                       count++;
```

下列為讀取 Land.txt 的方式，做法跟讀取 Character.txt 是一樣的，只差在 tempArray 的陣列大小不一樣。

```
109 //-------------------------------------Land
110             FileReader fr2 = new FileReader(filename2);
111             BufferedReader br2 = new BufferedReader(fr2);
112             String line2,tempstring2;
113             tempArray2= new String[2];
114             myList2 = new ArrayList<Integer>();
115             int iii=0;
116             String fline2=br2.readLine();
117             while((line2 = br2.readLine())!=null)
118              {
119                   tempstring2 = line2;
120
121                  tempArray2 = tempstring2.split(",");
122
123                for(iii=0;iii< tempArray2.length;iii++)
124                   {
125                       myList2.add(Integer.parseInt(tempArray2[iii]));

127                   }
128
129              }
130           System.out.println(myList2);
131           br2.close();
132
133           k2=myList2.size()/2;
134           int countt=0;
135           trans_array2 = new Integer[k2][5];
136           for(int x=0;x<myList2.size()/2;x++)
137           {
138
139            for(int y=0;y<2;y++)
140               {
141                   trans_array2[x][y]=myList2.get(countt);
142                   countt++;
```

- **儲存功能：**

以一個 static method，名稱為 Save()來實現儲存功能，需傳入 String 型態的檔案名稱(filename)，要傳兩個，到 Save 裡，並且讓 Save throws IOException 當發生異常時拋出錯誤訊息。Save 內部說明如下：

(1) 宣告一個 OutputStreamWriter 物件取名為 output1。將 new FileOutputStream(filename)傳入 output1 的建構式裡讓 output1 知道要寫入的對象檔案名稱，並且傳入第二個參數 UTF-8 讓寫入的編碼方式以 UTF-8 來進行。

(2) 開始寫入檔案，先直接寫入第一行「Round: 變數值,Turn: 變數值」。

(3) 再來利用雙層 for 迴圈寫入角色遊戲數值，寫的方式為先用 output1.write()將 trans_array[][](在主程式中會將 myList 的值存到 trans_array[][])裡的值寫回去檔案裡，每當判斷到寫入的對象為 IMAGE_FILENAME 時，會多寫入一個\n 來換行，保持和原本檔案的格式一樣。完成寫入後將 output1 關掉。

```
149    public static void Save(String filename, String filename2) throws IOException {
150        //// TODO: You should save the changed variables into original data (filename). ////
151        OutputStreamWriter output1 = new OutputStreamWriter(new FileOutputStream(filename),"UTF-8")
152        output1.write("Round:"+Numround+","+"Turn:"+trans_array[act][1]+"\n");
153        for(int x=0;x<myList.size()/5;x++)
154           {
155               for(int y=0;y<5;y++)
156               {
157             output1.write(trans_array[x][y]);
158
159            if(y==4) {
160                output1.write("\n");
161                continue;
162             }
163             output1.write(",");
164         }
```

(4) 下列為儲存 Land.txt 檔內容的方式，和儲存 Character.txt 的觀念一樣利用 write()將值寫入。

```
172        OutputStreamWriter output2 = new OutputStreamWriter(new FileOutputStream(filename2),"UTF-8");
173        output2.write("LOCATION_NUMBER, owner"+"\n");
174        for(int x=0;x<myList2.size()/2;x++)
175           {
176               for(int y=0;y<2;y++)
177               {
178            System.out.println(trans_array2);
179            output2.write(Integer.toString(trans_array2[x][y]));
180
181             if(y==1) {
182                output2.write("\n");
183                continue;
184             }
185             output2.write(",");
186         }
187
188     }
```

- **Random 方法：**

Random()需要實現的功能，像是角色之間的切換、回合數值增加、顯示目前的玩家。

先提醒待會此方法中經常出現的變數的意思：

int act 是用來判斷輪到哪位玩家的計數器。int Numround 用來記錄回合數。

程式說明如下：

用 for 迴圈並透過計數器 counter 找出第一個可以玩的人是誰(status>0)，找到後 break 跳出，進入下方的判斷式。

```
146        int count=0;
147        for(int i=0;i<A1063329_Checkpoint4.myList.size()/5;i++) {
148            count++;
149            if(Integer.parseInt(A1063329_Checkpoint4.trans_array[i][3])>0) {
150                //role2.setText("Turn Character "+A1063329_Checkpoint4.trans_arra
151                act=i;
152                break;
153            }
154            if(count==4) {
155                //role2.setText("Turn Character "+A1063329_Checkpoint4.trans_arra
156                break;
157            }
158
159        }
```

while 迴圈結束後進入另一個 if 判斷式，此判斷是判斷 status 是否不等於零，當條件成

立時，先將全域變數 randomNum 設成亂數值，作法為 Math.random()*6+1，由於 Math.random 是一個 0 到 1 的小數，因此乘以 6+1 並轉成整數型態才能做出骰子的效果。接著將目前輪到哪位玩家的字串顯示在 role2 這個 label 上，哪位玩家用玩家編號表示(trans_array[art][1])，接著把該位玩家的 location 值加上 randomNum(也就是他擲出的點數)和把他的 status 減一代表他玩完，由於 trans_array 為 String 型態，要先用 Integer.parseInt()轉成 int 才能夠做加減運算，做完運算後再把新結果用 Integer.toString()轉回去 String 型態存回去 trans_array 裡。

```
159 if(Integer.parseInt(trans_array[act][3])!=0) {
160
161     randomNum=(int)(Math.random()*6+1);
162     A1063329_GUI.role2.setText("Turn Character "+trans_array[act][1]);
163             trans_array[act][0]=Integer.toString(Integer.parseInt(trans_array[act][0])+randomNum);
164             trans_array[act][3]=Integer.toString(Integer.parseInt(trans_array[act][3])-1);
```

另一個 if 判斷式為判斷 act 是否為 4，判斷是不是輪到第四位玩家，是的話用 for 迴圈將所有人的 status 加 1，因為代表遊戲即將進入下一個新回合，如果 act 不等於 4 就把 act++，代表準備換下一位玩家，當下一位玩家點擊骰子後便會用 act++後的值進入上述的各種判斷式，重複上述的動作。

```
169             if(act==3) {
170                 for(int x=0;x<=act;x++)
171                 {
172                     trans_array[x][3]=Integer.toString(Integer.parseInt(trans_array[x][3])+1);
173                 }
174
175                 Numround++;
176                 int Numround2=Numround;
177                 A1063329_GUI.round2.setText("");
178                 A1063329_GUI.round2.setText("Round "+Integer.toString(Numround2));
179                 System.out.print("hi");
180
181
182                 act=0;
183                 A1063329_GUI.role2.setText("Turn Character "+trans_array[act][1]);
```

中間 else if 裡有個 while 迴圈，該迴圈用來幫助判斷跳過，檢查每個人的 status，若大於 0 則跳出迴圈，若判斷到小於零且為第四位玩家的畫，進行和上面 act==3 一樣的操作。

```
186             else if(act!=3){
187
188                 act++;
189                 int x=0;
190                 int y=0;
191                 y=act;
192                 System.out.println("y="+y+" trans_array="+trans_array[y][3]);
193                 //remeber before trans_array[0][1]status-1
194                 while(y<4)
195                 {
196                     //System.out.println("while's y = "+y+" trans_array"+trans
197                     if(Integer.parseInt(trans_array[y][3])>0)
198                     {
```

```
200                     x=y;
201                     break;
202
203
204                 }else if((Integer.parseInt(trans_array[y][3])<=0) && y==3)
205                     {
206                         x=0;
207                         System.out.println("enter litte while when y=3");
208                         Numround++;
209                         A1063329_GUI.round2.setText("Round "+Numround);
210                         for(int d=0;d<k;d++)
211                         {
212                             trans_array[d][3]=Integer.toString(Integer.parseInt(trans_array[d][3])+1);
213                             System.out.println("大家加一:"+trans_array[d][3]);
214                         }
215                         act=0;
```

剛剛講的都是當最外圈的 if 條件成立時所進行的動作，若外圈的 if 條件不成立時，則進入 else if 的判斷式，判斷條件為 Integer.parseInt(trans_array[act][3])==0 和 act==k-1

意思是 status 為零和 act 等於 3 是否同時成立，換句話說就是判斷第四位玩家是否 status 為零，條件成立的話先用 for 迴圈將每一位玩家的 status 加 1，然後將 act 歸零，因為第四位玩家同時為最後一位玩家代表回合將要結束，所以將 act 歸零以利下個回合的開始。若 else if 的條件仍不成立，則進入 else 裡的動作，讓 role2 顯示目前輪到誰(根據 act 目前的數值)，再把 act++，來準備判斷下一位玩家的狀況。

```
183
184         }else if(Integer.parseInt(trans_array[act][3])==0 && act==k-1){
185
186                 for(int x=0;x<=act;x++)
187                 {
188                     trans_array[x][3]=Integer.toString(Integer.parseInt(trans_array[x][3])+1);
189                 }
190                 act=0;
191
192         }else{
193               A1063329_GUI.role2.setText("Turn Character "+trans_array[act][1]);
194                 act++;
195
196               }
197     }
198
```

使用說明：

- ✓ 程式開始執行後，會先和資料庫建立連線，玩家可以看到大富翁遊戲畫面，並開始進行遊戲。玩家可點擊中間圓形按鈕 dice 來進行擲骰子的動作，按下 dice 後可達鴨開始根據點數進行移動，而若有玩家的 status 小於等於 0 則跳過，鴨子也隨之禁止不動。擲出的數值會顯示在右邊的框框裡，框框下方可以看到下一位玩家的玩家編號。當玩家玩到一個段落想要存檔的話，可點擊畫面左上方的 Save 按鈕來進行存檔，玩家可打開附件中的 Character.txt 檔來確認是否有存檔成功，接著若要結束遊戲的話，可點擊畫面右下方的 Exit 按鈕來結束遊戲、關閉視窗。當玩家再次執行遊戲，並且想要從上次結束時的數據接續著玩的話，可點擊 Save 旁邊的 Load 按鈕，程式會讀取上次存檔的紀錄，顯示上次的回合數和下一個該誰玩，等畫面閃爍一下後玩家便可開始繼續玩遊戲。遊戲畫面可參考圖片(如下圖)。
- ✓ 當鴨子經過土地時會根據土地狀況跳出對應的視窗，會詢問玩家是否購買土地或是要求玩家繳交過路費，當玩家購買土地後即成為地主，地上會顯示自己的編號，當玩家錢不

夠買時便無法購買土地，但仍需繳交過路費(即時錢為負數)，而每通過一次起點即可獲得兩千元。