OOP 期末專題

Final Project

主題：NBA・BOMB

作者

108 級電機 B 班 0410746 林志恩

108 級電機 B 班 0410766 陳胤銓

一、題目描述

NBA 炸彈超人 – 改良版 NBA・BOMB

我們的構想是將大學生關注的 NBA 籃球運動融入炸彈超人的遊戲之中。
我們製作了人物選單，並事先選擇了十位 NBA 的籃球巨星，讓玩家可以從中選擇喜愛的角色遊玩本遊戲。此外，許多的背景圖也是利用像是 Kobe Bryant、Michael Jordan、LeBron James 的經典照片，讓遊戲充滿了 NBA 的氛圍。

二、分工及每人負責的部分

林志恩:

產生隨機地圖 --- O
計時器 --- O (利用計時器來製作 player 的行走控制、炸彈爆炸倒數、火焰消失倒數、閃爍倒數、PUSH、Ready 倒數等等)
放置炸彈 --- O
炸彈爆裂 --- O
產生特殊武器與道具 --- O
使用特殊武器與道具 --- O
角色設計 --- O (有十位不同的 NBA 球員可供選擇)
背景圖片與動畫 --- O (人物的大圖與小圖、每個 panel 的背景圖片，都經過修圖美化並控制了大小)
Debug
程式優化

陳胤銓:

判斷鍵盤輸入 (2 個 player 的方向鍵與炸彈鍵) --- O
遊戲控制介面 (排版與選項) --- O (分成起始畫面、選角畫面、準備畫面、遊戲畫面與排行榜)
電腦玩家行走路徑 (Artificial Intelligence) --- O (使用老鼠走迷宮的方法)
電腦玩家移動 --- O
電腦玩家放置炸彈 --- O
判斷輸贏 --- O (最後贏家必須為一位 Player1 或 Player2，如果只剩下電腦玩家則輸掉遊戲)
設計關卡 (不同的遊戲模式) --- O (進入警告的倒數時間，地圖會慢慢自動封閉，讓玩家的移動區域縮小)
排行榜 --- O (開檔案，可點擊 Rank 觀看紀錄)
Debug
程式優化

三、使用的 IDE

Visual C#

四、系統架構和分析

有八個重點是我們自己原創的程式寫法，以下會一一說明：

1. Character (panel)：

點擊 Start 之後，panel 就會切換到 Character 顯示在 form 中。我利用 characterset 把十個選角色用的 picturebox 初始圖片設好，並排到適當位置。一開始預設 player1 和 player2 選擇中間的角色，利用 picturebox 的 paint 在外圍畫出框框 (player1：紅色、player2：藍色)，並利用 timer 讓框框的顏色閃爍，讓玩家知道現在是待選擇的狀態。經過 keydown 事件(和遊玩遊戲時一樣的操作方式，player1 移動 W-A-S-D、選擇 Space；player2 移動上-下-左-右、選擇 L)，改變 selectindex1 與 selectindex2 並經過換算，讓 index 所在的 picturebox 外框顯示紅色或藍色。在選角的過程中，會判斷是否下一項已被另外一個玩家選擇而跳過，因此兩位玩家並不會停留在同一角色，也不會選到一樣的角色。在 keydown 按下選擇鍵(L or Space)，那格的 picturebox 外框會停止閃爍，讓玩家知道已選擇此角色，如果另外一位玩家還沒選好角色，再按一次選擇鍵可以取消此角色去選其他角色。我們有找了 NBA 角色的大圖讓畫面更為豐富，也在旁邊加了 label 和一些圖示直接讓玩家瞭解如何操作。兩位玩家選完角色之後，兩位電腦玩家(CPU)會從剩下的八位角色隨機選取角色，而且不會重複選到同一角色。

2. Ready(panel)：

將玩家在 Character 選好的角色加上自動選好的電腦角色，一起顯示出來，並在中間說明等一下遊戲裡會出現的道具與其效果，讓玩家更容易上手這個遊戲。下方則顯示 Loading 並倒數 5 秒鐘。

3. 玩家移動：

Player1 移動 W-A-S-D (上-下-左-右);

Player2 移動 上-下-左-右;

AI 移動將在 AI 的部分做說明;

以 player1 來進行說明，當玩家按下 W-A-S-D 其中一個按鍵時觸發 form1 的 keydown 事件 W-A-S-D (為了避免斜向移動，如果 uptimer、downtimer、lefttimer、righttimer、timer1 的 enabled 都是 false，才進行以下步驟) 分別使 uptimer、downtimer、lefttimer、righttimer 四個方向的移動 timer 啟動，四個方向的 timer 經過一個週期(以向上為例)：預設是 0.002 秒會進行一次座標更新。

如果沒有撞到障礙物(判斷上方 map 的 tag 所存的值)：

player1.location = new point (player1.location.x,player1.location.y-step);

因為 player1 是 picturebox，地圖則是用 Game_paint 事件畫出來的，所以 picturebox 移動的時候會自動觸發 Game_paint，為了避免 lag 所以我們使用部分重劃的技巧 (將在 Game_paint 進行說明)

停止移動:

(1) 撞到障礙物

(2) W 按鍵放開，觸發 keyup 事件把 uptimer 關掉並啟動 timer1(將 player1 移動到格子上面，避免人物處於格子之間) --> 關掉 timer1

4. 產生地圖：

我們先將地圖的中間保留一個九宮格並設成通路，也將四個角落的角色起始位置設成通路。利用四次的 for 迴圈，先從地圖左上角的位置往九宮格的最左上角移動，計算好往右與往下還差幾步，隨機往四個方向移動一步，再改變往右和往下還差幾步的值，一直到兩者皆為 0。其他三次 for 迴圈也為同理，而超界的部分我們也有做判斷，經過多次嘗試後完全沒有問題。再來我們就讓不能被炸的牆壁產生在不是通路的地方，這樣就不會擋到玩家，也不會讓玩家因此不能進行遊戲；而可以被炸的樹則是最後隨機產生在剩下的位置，我們有控制好牆壁和樹產生的數量與比例，並讓可以被炸的樹木裡有一定機率隱藏了特殊道具，只要玩家把樹炸開，就會顯示出來。

5. AI(CPU)：

在遊戲開始的時候啟動 aitimer 分別呼叫 AI(0)和 AI(1) (為 player3 和 player4)，在 AI 函數中先呼叫 aimapanalyze 函數建構出 AI 目前能到達的地方存在 aimap[i,j].pass 裡面，在使用 bombdanger 將可到達且會被炸彈炸到的地方 aimap[i,j].pass 值改成 2，再呼叫 aimapvalue 函數使用巢狀回圈計算每一個地圖位置的 aimap[i,j].value 的值(不會到的地方 value 為負，會被炸到 value 會減小，能夠炸到樹或是人的地方 value 增加)，再來用巢狀回圈掃過 15*15 的 aimap[i,j].value 找出最大值得地方，存成 goali, goalj(AI 的目的地)，再來呼叫 ratwalk 函數使用老鼠走迷宮的方法(先宣告出一個 15x15 的二維陣列，將值都歸零，不能走的地方為-1，將所在位置的值+1 之後，向四周最小值的地方走，每走一走就存到路徑陣列裡面，當重複經過同依個地點時，會把多餘的路徑覆寫掉，變成最精簡的路徑)將路徑存成陣列，在 aitimer 下一次觸發的時候朝著路徑移動，當到達目的地時，放置炸彈，再次呼叫 AI(0)，重新建構路徑。

6. 重劃:

在 Game_paint 裡面多寫了一些判斷式，只有剛進到遊戲介面的時候會全部畫一次，接下來判斷哪一個角色在移動，重劃角色周圍九宮格，而不是全部重劃，讓畫面能更流暢。

7. Warning (警告時間)兩分鐘以內:
地圖從左下角開始一格一格填滿，逆時針繞兩圈，當人物被地圖壓到時，血量就會直接歸零，從遊戲裡出局。先設定一個方向變數和要填滿的座標：當 t%10==0 也就是經過一秒鐘的時候呼叫 fillin()函數，使用 switch(方向變數)更新要填滿的座標(當座標通過 x=y 這條直線，把方向變數從向右改成向上)，再呼叫 Game_paint 事件把紅色磚頭劃出來。

8. Rank (panel)：
在遊戲結束後，如果玩家獲得勝利，將會動態產生一個 Inputbox 給玩家輸入名字，按下 OK 之後新增或開啟在 Rank 資料夾下的檔案寫入名字、playnum、生命值、炸彈數、花費時間。在切換 Rank 介面的時候把資料從檔案讀出，存成結構陣列，再把結構陣列依照花費時間進行排序 (sort)，並輸出到 Rank 裡面的 label 上顯示出來。

函數原型(共 6 個檔案)

1. Ready_function.cs
private void readyset() //設定 Ready panel 的位置、picturebox 圖片設置

2. Rank.cs
public void fileopen(int playi, int life, int bomb, int lefttime, string namepass)//寫入檔案
public void fileread()//從檔案讀取
public void ranksort(int n)//將讀取的結構陣列依照時間排序

3. Inputbox.cs
private InputBox()//初始化
public String getValue()//傳回名字
public static bool Show(String title, String inputTips, bool isPassword, ref String value)//產生輸入視窗
private void Ok_Click(object sender, EventArgs e)//確認名字輸入
private void Cancel_Click(object sender, EventArgs e)//取消輸入

4. Game_function.cs
private void setimage()//設定圖片
private void setvalue()//設定初值
private void generate_map()//產生隨機地圖
public void bombvalue(int playernum, int x, int y, ref int index)//產生炸彈並設初值
public void bombsplash(int i, int j, int mapi, int mapj)//計算炸彈爆炸範圍
public void firetoroad(int i, int j, int mapi, int mapj)//把火焰變成一般路面

public void uirefresh()//更新介面
public void player1_specialitem(int playernum, int item)//角色 1 撿起特殊道具
public void player2_specialitem(int playernum, int item)//角色 2 撿起特殊道具
public bool iswaymove(int playno, int index, int direction)//判斷炸彈能不能被推動
private void AI(int aino)//用來呼叫有關 ai 的函數
public void aimapanalyze(int aino,int i,int j)//用遞迴建構出 ai 可到的地圖
public void bombdanger(int aino)//偵測地圖上哪個點會被炸到
public void aimapvalue(int aino)//設定 aimap.value 的值
public bool ratwalk(int aino,int aiy,int aix,int goaly,int goalx))//ai 走的路徑(老鼠走迷宮)
public void detectbomb(int aino)//ai 偵測周圍炸彈剩餘秒數
public bool setbomb(int aino)//判斷要不要放炸彈(AI)
public void isdied()//判斷遊戲勝負
public void gameclose()//關閉遊戲
public void fillmap()//warning 時間 將地圖縮小

5. Form1.cs
public Form1()
private void Form1_Load(object sender, EventArgs e)
private void button1_Click(object sender, EventArgs e) //Start 按鈕，切換到 Character 的 panel
private void button2_Click(object sender, EventArgs e) //Rank 按鈕，切換到 Rank 的 panel
private void Game_Paint(object sender, PaintEventArgs e)//控制遊戲畫面重劃
private void uptimer_Tick(object sender, EventArgs e)//控制往上移動
private void downtimer_Tick(object sender, EventArgs e))//控制往下移動
private void righttimer_Tick(object sender, EventArgs e)//控制往右移動
private void lefttimer_Tick(object sender, EventArgs e)//控制往左移動
private void timer1_Tick(object sender, EventArgs e)//將角色修正到正確位置
private void uptimer2_Tick(object sender, EventArgs e)//控制往上移動
private void downtimer2_Tick(object sender, EventArgs e))//控制往下移動
private void righttimer2_Tick(object sender, EventArgs e))//控制往右移動
private void lefttimer2_Tick(object sender, EventArgs e))//控制往左移動
private void timer2_Tick(object sender, EventArgs e)//將角色修正到正確位置
private void Form1_KeyUp(object sender, KeyEventArgs e)//偵測鍵盤放開
private void Form1_KeyDown(object sender, KeyEventArgs e)//偵測鍵盤按下
private void bombtimer_Tick(object sender, EventArgs e)//計算炸彈存在時間
private void mutekitimer_Tick(object sender, EventArgs e) //控制被火焰炸到後的無敵時間(Player 1)

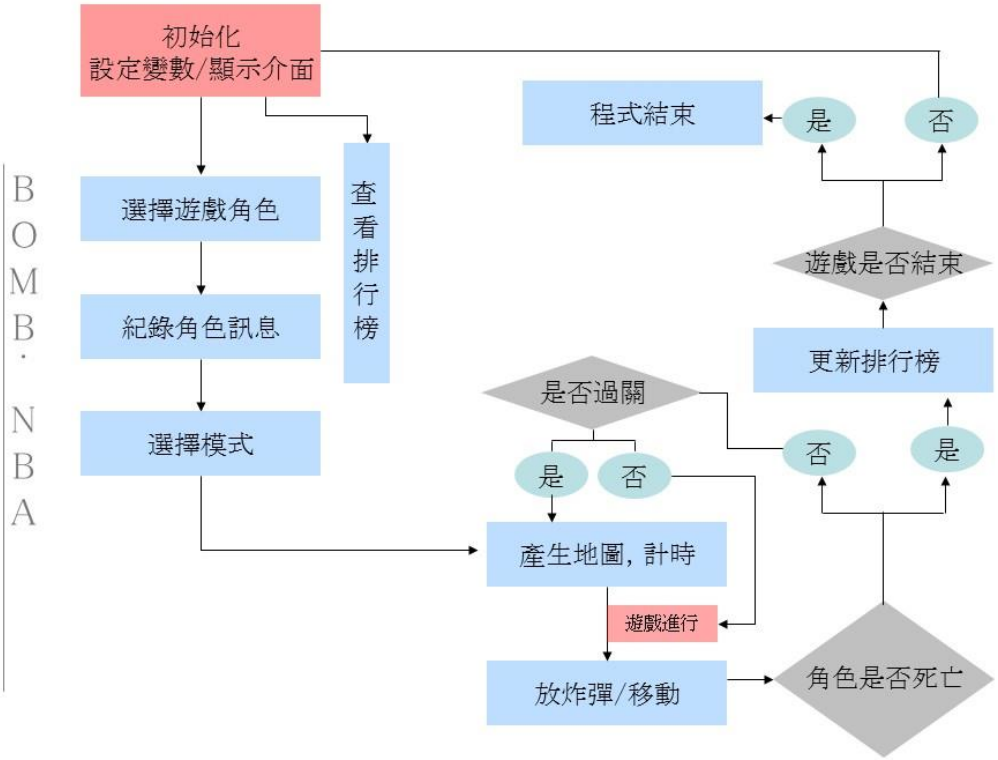
private void mutekitimer2_Tick(object sender, EventArgs e) //控制被火焰炸到後的無敵時間(Player 2)
private void mutekitimer3_Tick(object sender, EventArgs e) //控制被火焰炸到後的無敵時間(CPU 1)
private void mutekitimer4_Tick(object sender, EventArgs e) //控制被火焰炸到後的無敵時間(CPU 2)
private void bombmovetimer_Tick(object sender, EventArgs e) //控制 PUSH 推炸彈的時間
private void aitimer_Tick(object sender, EventArgs e)//控制 ai 移動
private void gametimer_Tick(object sender, EventArgs e)//記算遊戲時間
private void warning_Paint(object sender, PaintEventArgs e)//劃出 warning 圖示
private void readytimer_Tick(object sender, EventArgs e) //控制 Ready panel 的維持時間，時間到進入 Game panel
private void button3_Click(object sender, EventArgs e) //從 Rank panel 返回 Main panel(主畫面)

6. Character.cs
private void characterset() //將 picturebox 初始圖片設置、位置設置
private void selecttimer_Tick(object sender, EventArgs e)//控制紅色與藍色外框的閃爍
private void Character_Paint(object sender, PaintEventArgs e)//讓十位角色的外框重劃
private void charactera3_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void characterb3_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void charactera1_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void charactera2_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void charactera4_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void charactera5_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void characterb1_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void characterb2_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色
private void characterb4_Paint(object sender, PaintEventArgs e) //分別重劃自己的外

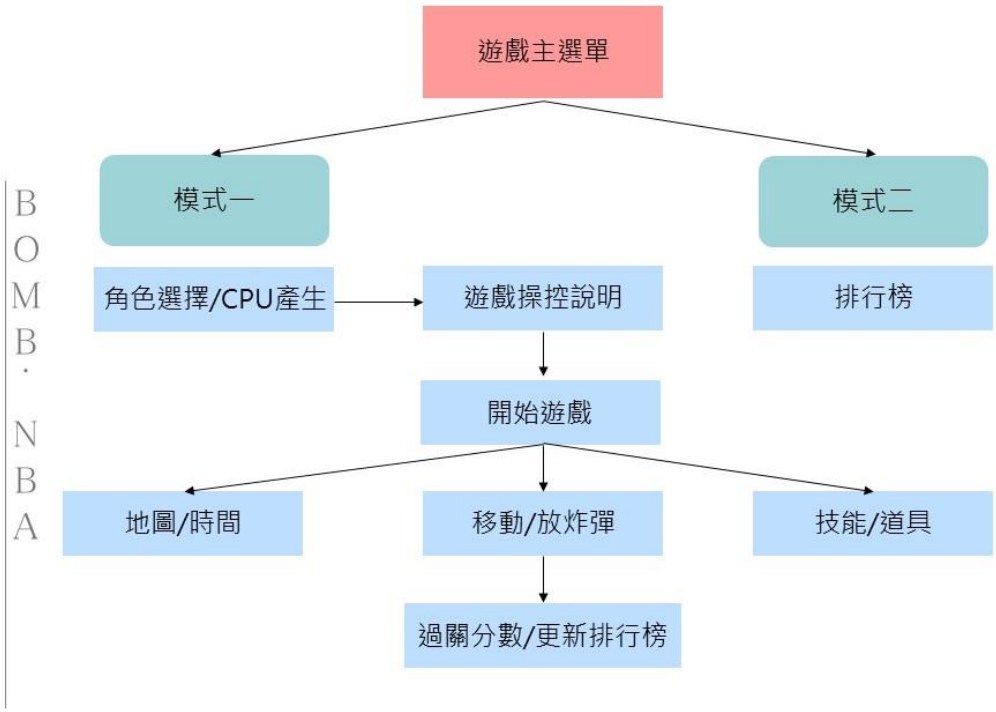
框、判斷為紅色或藍色或無顏色

private void characterb5_Paint(object sender, PaintEventArgs e) //分別重劃自己的外框、判斷為紅色或藍色或無顏色

流程圖：



架構：

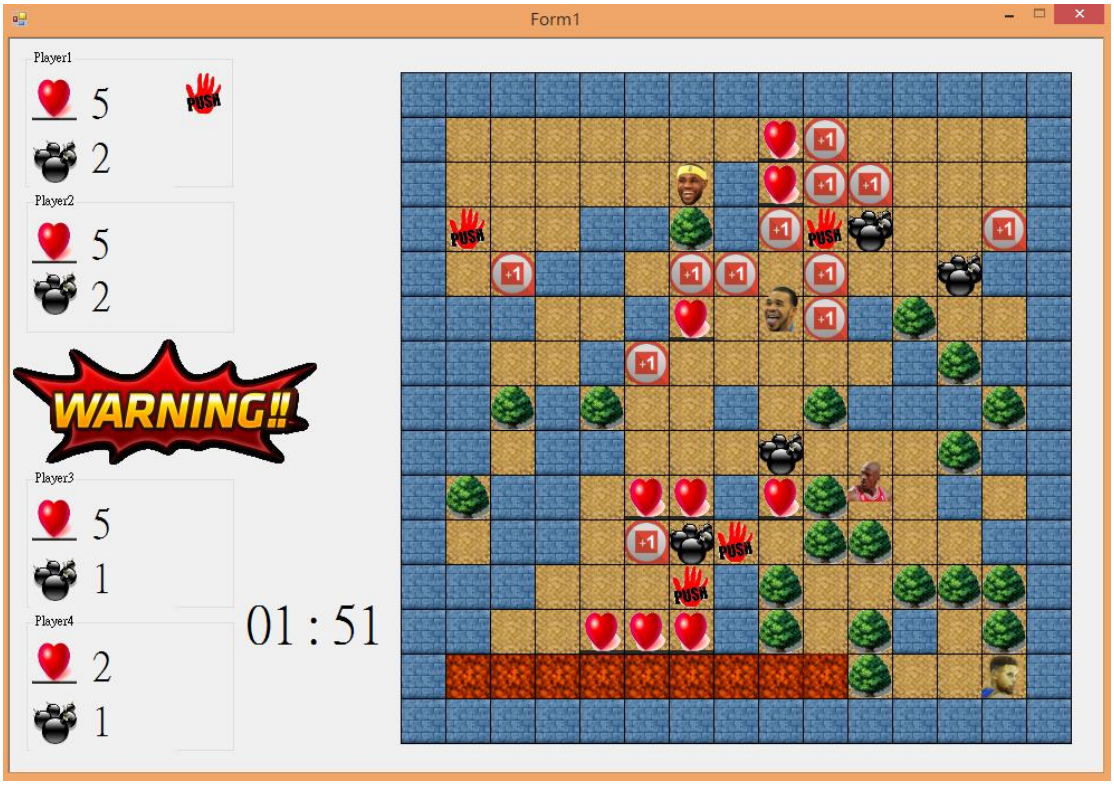

遊戲主選單
模式一
模式二
角色選擇/CPU產生
遊戲操控說明
排行榜
開始遊戲
地圖/時間
移動/放炸彈
技能/道具
過關分數/更新排行榜
BOMB·NBA

五、執行結果


Form1
NBA·BOMB
Start
Rank
THE RETURN
BRYANT
24


Form1
Player 1
V. S.
Player 2
LeBron James
Stephen Curry
W
A
S
D
Press Space to be ready
Press L to be ready


Form1
Player 1
CPU
生命值+1
炸彈數+1
爆炸距離+1
推炸彈模式
CPU
Player 2
Loading . . .
3

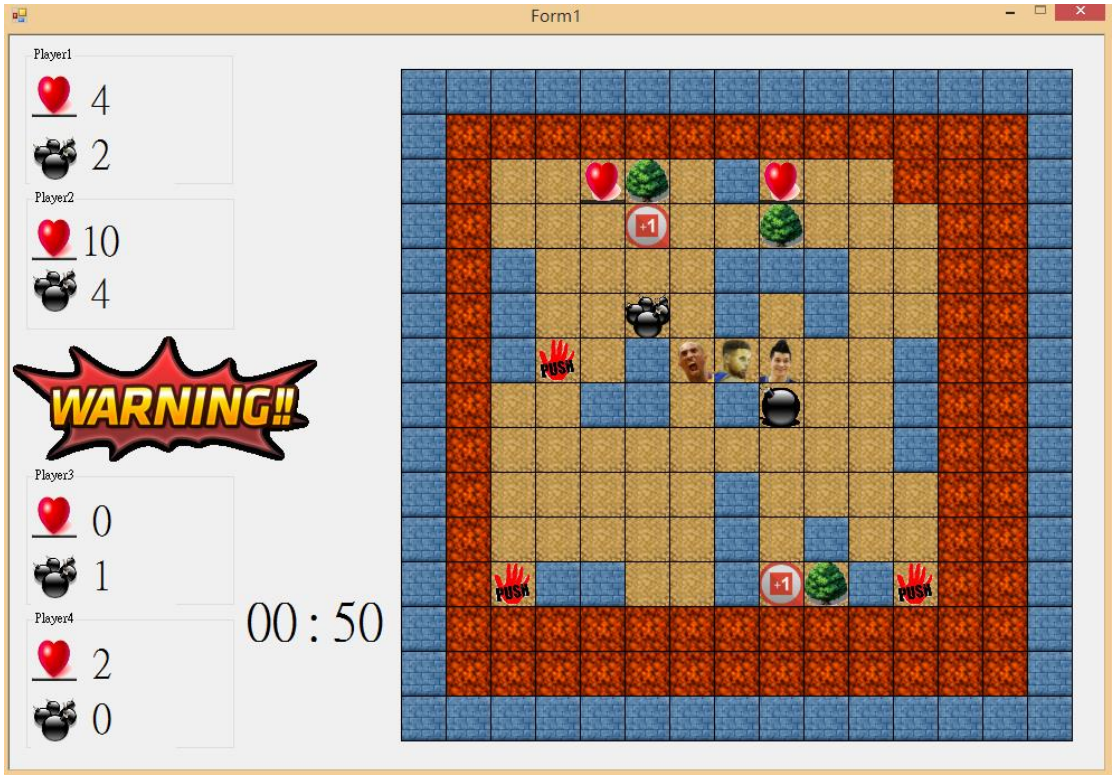

Form1
Player1
5
2
Player2
5
2
WARNING!!
Player3
5
1
Player4
2
1
01 : 51

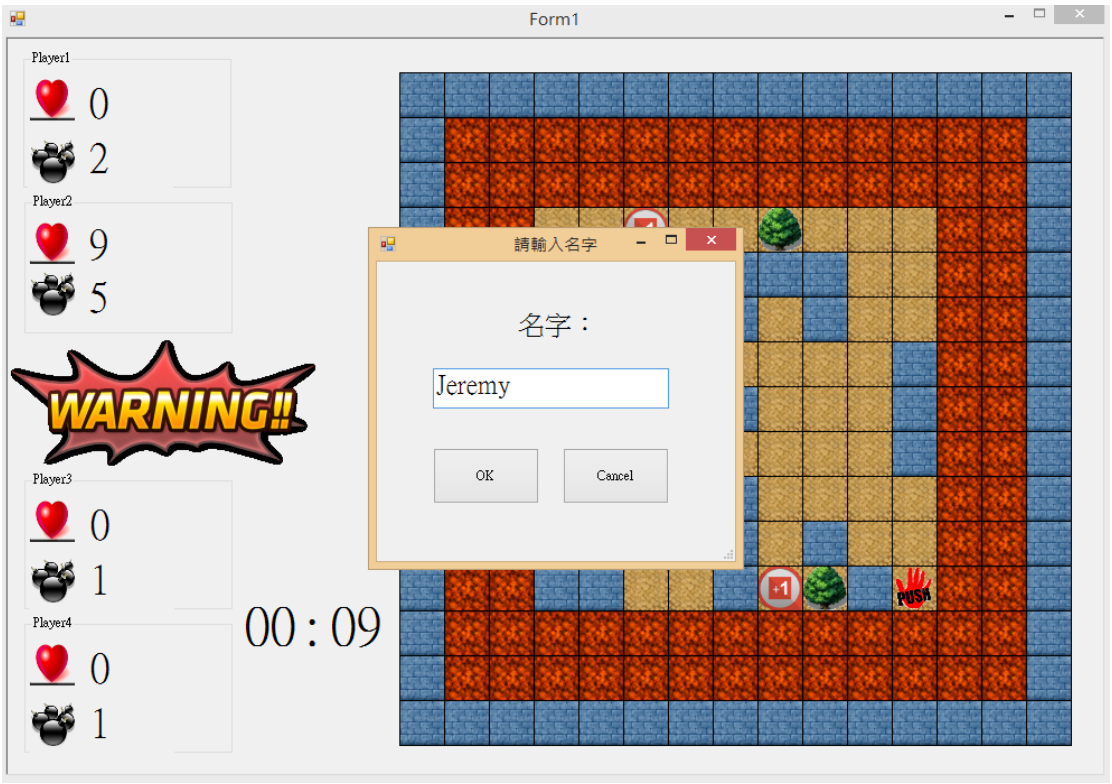

Form1
Player1
2
2
Player2
0
2
WARNING!!
Player3
5
1
Player4
0
1
01 : 26
請輸入名字
名字：
Jacky
OK
Cancel


Form1
Name
playernumber
life
bomb
usetime
Jacky
1
2
2
01 : 34
Back Main


Form1
Player 1
V. S.
Player 2
HORNETS
7
LAKERS
24
Jeremy Lin
Kobe Bryant
W
A
S
D
Press Space to cancel
Press L to be ready


Form1
Player 1
CPU
生命值+1
炸彈數+1
爆炸距離+1
CPU
Player 2
推炸彈模式
Loading . . .
1


Form1
Player1
4
2
Player2
10
4
WARNING!!
Player3
0
1
Player4
2
0
00 : 50


Form1
Player1
0
2
Player2
9
5
WARNING!!
Player3
0
1
Player4
0
1
00 : 09
請輸入名字
名字：
Jeremy
OK
Cancel
+1
PUSH


Form1
Name playernumber life bomb usetime
Jacky 1 2 2 01 : 34
Jeremy 2 9 5 02 : 51
Back Main

六、心得報告: 遇到的問題，如何解決或結果分析

（一）問題與解決方式：

1. 在產生人物選單時發現會有一個 picturebox 異常變色，經過多次測試和觀察，推測出應該是指標指向物件的部分寫錯了，之後經過修正就變正常了，並使我們再次仔細檢查。

2. 鍵盤控制一開始是寫在 keydown 事件裡面，每一下鍵盤敲擊都會觸發移動，當放開的時候啟動 timer1 移動到正確的格子上，但是這個方法會讓速度不一致，而且程式碼會有冗長的判斷，且畫面看起來卡卡的。經過思考後，我們決定使用 keydown 觸發四個方向的移動 timer，keyup 的時候觸發 timer1 移動到正確的格子上，這樣不但讓程式少了 30 行，而且畫面更順暢更容易管理。

3. 一開始我們使用 15*15 的 picturebox 當作地圖，但是會導致人物以及炸彈圖片的透明化很難處理，後來把地圖直接畫在 panel 上面，人物以 picturebox 來顯示，接下來只要把 picturebox.background 設成 transparent 就可以了，但是這樣反而會讓人物移動的時候變 lag，會有閃爍的問題。

4. 閃爍的問題:

之前人物畫在 panel 上後，我們發現人物在行走上會有閃爍的問題，我們利用

在上機時，助教用心教大家的”DoubleBuffered”來解決。但問題在於，我們作畫的 panel 並沒有屬性讓我們直接使用，而是透過上網查詢大量資訊並一步步證實，讓我們得知在 Form1 加上少許程式碼，便能讓我們遊戲畫面的 panel 擁有”DoubleBuffered”的效果，也因而解決了人物在行走上的閃爍問題。
程式碼如下：

```
if (System.Windows.Forms.SystemInformation.TerminalServerSession)
return;

System.Reflection.PropertyInfo aProp =
typeof(System.Windows.Forms.Control).GetProperty(
"DoubleBuffered",
System.Reflection.BindingFlags.NonPublic |
System.Reflection.BindingFlags.Instance);

aProp.SetValue(Character, true, null);
aProp.SetValue(Ready, true, null);
aProp.SetValue(Game, true, null);
aProp.SetValue(warning, true, null);
```

5. Keydown 事件的問題:
之前想要把 Player2 放置炸彈鍵設置為 Enter，可是 Keydown 無法偵測到 Enter 按鍵，推測可能是因為有其他控件產生的干擾。詢問助教後，發現其實只要另外使用別的按鍵就行，不必拘泥在這種小地方，因此我們最後決定使用按鍵 L。

6. 預測 AI 路徑的時候，使用遞迴產生路徑，一直遇到陣列超界的問題。
   → 使用老鼠走迷宮的方法，使用 while 迴圈處理就 OK 了。

7. 推炸彈的時候畫面會 lag 。
   → 使用部分重劃就解決了這個問題，不必讓整張地圖重劃。

8. AI 移動會 lag。
   → 使用部分重劃就解決了這個問題，不必讓整張地圖重劃。

9. 開檔的時候會當掉。
   → 把檔案刪掉重建就 OK。

10. Rank 排版會有問題，容易亂掉。
   → 分成多個 label 比較整齊。

（二）心得分享：

林志恩：

這次的 OOP 期末專題真的是經歷了重重考驗，從一開始構思到開始製作，常常覺得是不是把題目訂得太難，有想要放棄的念頭，但是我們熬過來了，不僅將我們的專題完成，寫了 3182 行的 code，也入選了決賽，最後拿到優等的佳績。這段路程收穫最大也學習最多的一定是我們自己，除了將上課所學的東西融會貫通，還要自己想辦法將 idea 付諸實行，遇到困難要懂得發問並找尋辦法解決。看到最後執行的結果，心裡真的是非常有成就感，想到一切的努力與辛苦終於化成甜美的果實，不禁潸然淚下，感動萬分。

陳胤銓：

這次的專題克服了很多困難，還有燃燒了許多的期末考成績總算順利的完成，也在決賽獲得了優等的佳績，雖然還有很多目標還沒有完成，兩個人一起移動的時候有機率會 lag，傳送點也尚未完成，期待在這個暑假能夠補完應該完成的進度，並且讓 AI 能夠更聰明有效率的去攻擊玩家。在寫排行榜的部份，才讓我第一次了解檔案原來是這樣應用的，目前是把資料全部讀到結構陣列裡面再排序輸出到螢幕上(效率不太好)，應該要另外建一個 index 的檔案來輔助使用增加效率，避免重複執行排序。

七、參考書籍

Google 等網路資訊
助教在 Lab 所教的內容與其上課的檔案
同學指點並自己融會貫通

諸如：panel 的 Doublebuffered 使用方法、重劃後背景的透明問題、picturebox 的 paint 方法(畫外框)等等。

八、Source code

**********Ready_function.cs**********

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

```
namespace NBA_BOMB
{
    public partial class Form1 : Form
    {
        int readyload; //5 秒
        private void readyset()
        {
            Ready.BackgroundImage = new
Bitmap(Properties.Resources.Ready_1000);
            readyload = 5;
            readytime.Text = readyload.ToString();
            readytimer.Enabled = true;
        }
    }

}
```

**********Rank.cs**********

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;

namespace NBA_BOMB
{
    public partial class Form1 : Form
    {
        struct ranktype
        {
            public string name;
            public int playnum;
            public int life;
```

```
        public int bomb;
        public int lefttime;
    };
    ranktype[] data = new ranktype[30];//可存 30 筆資料
    public void fileopen(int playi, int life, int bomb, int lefttime, string namepass)
    {
        if (namepass == "")
            return;
        BinaryWriter outFile = new
BinaryWriter(File.Open("..//..//Rank//Rank.dat", FileMode.Append));
        outFile.Write(namepass); // 寫入 字串
        outFile.Write(playi);   // 寫入 整數
        outFile.Write(life);    // 寫入 整數
        outFile.Write(bomb);    // 寫入 整數
        outFile.Write(lefttime);    // 寫入 整數

        outFile.Close(); // 關閉檔案
    }
    public void fileread()
    {
        Ranktitle.Text = "Name              playernumber      life      bomb
usetime\n\n";// 文字方塊 先清空
        Rankname.Text = "";
        ranklabel.Text = "";
        if (!File.Exists("..//..//Rank//Rank.dat"))
            return;

        BinaryReader inFile = new
BinaryReader(File.Open("..//..//Rank//Rank.dat", FileMode.Open));

        int rankindex = 0;
        /*String my_name;
        int my_playi;
        int my_life;
        int my_bomb;
        int my_lefttime;*/
```

```
            while (inFile.BaseStream.Position < inFile.BaseStream.Length) // 傳回
下一個可供使用的字元，但不消耗它
            {
                data[rankindex].name = inFile.ReadString(); // 讀出 字串
                data[rankindex].playnum = inFile.ReadInt32();
                data[rankindex].life = inFile.ReadInt32();
                data[rankindex].bomb = inFile.ReadInt32();
                data[rankindex].lefttime = inFile.ReadInt32();

                rankindex++;
            }

            ranksort(rankindex);
            ranklabel.Text = "";
            Rankname.Text = "";
            int min;
            int sec;
            for (int i = 0; i < rankindex; i++)
            {
                Rankname.Text = Rankname.Text + data[i].name + "\n";

                ranklabel.Text = ranklabel.Text + (data[i].playnum + 1).ToString() +
"                 ";
                ranklabel.Text = ranklabel.Text + data[i].life.ToString() + "
";
                ranklabel.Text = ranklabel.Text + data[i].bomb.ToString() + "
";
                min = data[i].lefttime / 60;
                sec = data[i].lefttime % 60;
                if (min < 10)
                    ranklabel.Text = ranklabel.Text + "0";
                ranklabel.Text = ranklabel.Text + min.ToString() + " ";
                ranklabel.Text = ranklabel.Text + ": ";
                if(sec<10)
                    ranklabel.Text = ranklabel.Text + "0";
                ranklabel.Text = ranklabel.Text + sec.ToString();

                ranklabel.Text = ranklabel.Text + Environment.NewLine;
```

```
            }

            inFile.Close(); //  關閉檔案
        }
        public void ranksort(int n)
        {
            ranktype tmp;
            for (int i = 0; i < n; i++)
            {
                for (int j = i + 1; j < n; j++)
                {
                    if (data[j].lefttime < data[i].lefttime)
                    {
                        tmp = data[i];
                        data[i] = data[j];
                        data[j] = tmp;
                    }
                }
            }
        }
    }
}
************************************* *****Inputbox.cs*************************************
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace NBA_BOMB
{
    public partial class InputBox : Form
    {
        private InputBox()
```

```
        {
            InitializeComponent();
        }
        public String getValue()
        {
            return textBox1.Text;
        }
        public static bool Show(String title, String inputTips, bool isPassword, ref
String value)
        {
            InputBox ib = new InputBox();
            if (title != null)
                ib.Text = title;
            if (inputTips != null)
                ib.label1.Text = inputTips;

            if (ib.ShowDialog() == DialogResult.OK)
            {
                value = ib.getValue();
                ib.Dispose();
                return true;
            }
            else
            {
                ib.Dispose();
                return true;
            }
        }

        private void Ok_Click(object sender, EventArgs e)
        {
            Form1.name = getValue();
            if (Form1.name != "")
                this.Close();
            else
                MessageBox.Show("空");
        }
```

```
        private void Cancel_Click(object sender, EventArgs e)
        {
            Form1.name = "";
            this.Close();
        }
    }
}
**************************************Game_function.cs**************************************
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace NBA_BOMB
{
    public partial class Form1 : Form
    {
        public static string name;
        Pen myPen = new Pen(Color.Black, 1);
        Bitmap scr = new Bitmap(1000, 700);
        PictureBox[] play = new PictureBox[4];

        const int num = 15;//17*17  地圖
        const int topy = 30;//左上角 y 值
        const int leftx = 350;//左上角 x 值
        const int mapwidth = 40;//寬度
        const int mapheight = 40;//高度
        const int step = 4;//控制移動速度(40 的因數)
        const int aistep = 4;//控制移動速度(40 的因數)
        const int bombnum = 10;
        const int bombattacknum = 2;
        const int mutekitimerload = 20; //被炸彈炸到的無敵時間長度
        const int pushtimerload = 50;
```

```
const int ainum = 2;
const int playnum = 2;
const int warnwidth = 272;
const int warnheight = 112;
const int warnt = 1200;//warning 倒數時間  120 秒
const int aiwaymax = 300;//aiway 陣列大小
Bitmap scr1 = new Bitmap(warnwidth, warnheight);
int t = 1800; //遊戲時間
int warindex = 0;
bool[] movingflag = new bool[ainum];
int[] aiindex = new int[ainum];
int btomi, btomj;
int[] aidirection = new int[ainum];
int warni, warnj, warndirection;
//int player1_x = 1;
//int player1_y = 1;

Image pass = new Bitmap(Properties.Resources.brownground);
Image trash = new Bitmap(Properties.Resources.tree4);
Image notpass = new Bitmap(Properties.Resources.icewall);
Image bomber = new Bitmap(Properties.Resources.bomb01);
Image fire = new Bitmap(Properties.Resources.onfire);
Image warnwall = new Bitmap(Properties.Resources.redground);
Image[] tool = new Image[10];
Image[] war = new Image[6];

Random fixRand = new Random();

public struct maptype
{
    public int x;
    public int y;
    public int tag;
    public int item;
    public bool fire;
    public int firetoroadflag;
    public int bombplayer;
    public int bomblocationindex;
```

```
        public bool warn;
};

public struct bombtype
{
        public int x;
        public int y;
        public int timesec;
        public int distance;
        public bool exist;
        public int onfire;
        public int bombdirection; //上：1；左：2；下：3；右：4
};

public struct playertype
{
        public int playernum;
        public int x;
        public int y;
        public int bombindex;
        public int bombdistance;
        public int bombcount;
        public int mutekitime;
        public int life;
        public int direction; //上：1；左：2；下：3；右：4
        public int pushtime;
};

public struct aimaptype
{
        public int pass;
        public int value;
        public int ratvalue;
        public int timesec;
};

public struct aiwaytype
{
```

```
        public int valuex;
        public int valuey;
};

public struct wayindextype
{
        public int index;
        public int success;
        public int direct;
        public int back;
};

maptype[,] map = new maptype[num, num];
bombtype[,] bomb = new bombtype[4, bombnum];
playertype[] player = new playertype[4];
aimaptype[,,] aimap = new aimaptype[ainum, num, num];
aiwaytype[,] aiway = new aiwaytype[2, aiwaymax];
wayindextype[] wayindex = new wayindextype[2];
private void setimage()
{
        tool[0] = new Bitmap(Properties.Resources.brownground);
        tool[1] = new Bitmap(Properties.Resources.bomb14);
        tool[2] = new Bitmap(Properties.Resources.love);
        tool[3] = new Bitmap(Properties.Resources.push);
        tool[4] = new Bitmap(Properties.Resources.distance);
        war[0] = new Bitmap(Properties.Resources.war1);
        war[1] = new Bitmap(Properties.Resources.war2);
        war[2] = new Bitmap(Properties.Resources.war3);
        war[3] = new Bitmap(Properties.Resources.war4);
        war[4] = new Bitmap(Properties.Resources.war5);
        war[5] = new Bitmap(Properties.Resources.war6);
}

private void setvalue()
{
        warni = 13;
        warnj = 0;
        warndirection = 4;//1:上,2:左,3:下,4:右
```

```
t = 1800;
play[0] = player1;
play[1] = player2;
play[2] = player3;
play[3] = player4;
for (int i = 0; i < 4; i++)
        play[i].Visible = true;

for (int i = 0; i < num; i++)
        for (int j = 0; j < num; j++)
        {
                map[i, j].x = leftx + j * mapwidth;
                map[i, j].y = topy + i * mapheight;
                map[i, j].tag = 0;
                map[i, j].item = 0; //沒有特殊道具
                map[i, j].fire = false;
                map[i, j].firetoroadflag = 0;
                map[i, j].warn = false;
        }

for (int i = 0; i < 4; i++)
{
        player[i].playernum = i + 1;
        player[i].bombindex = 0;
        player[i].bombdistance = 1;
        if (i >= 2)
                player[i].bombcount = 1;
        else
                player[i].bombcount = bombattacknum;

        player[i].life = 5;
        player[i].mutekitime = mutekitimerload;
        player[i].pushtime = 0;
        switch (i)
        {
                case 0:
                        player[i].x = 1;
                        player[i].y = 1;
```

```
                break;
            case 1:
                player[i].x = 13;
                player[i].y = 13;
                break;
            case 2:
                player[i].x = 13;
                player[i].y = 1;
                break;
            case 3:
                player[i].x = 1;
                player[i].y = 13;
                break;
            default:
                break;
        }
    }

    for (int i = 0; i < 4; i++)
        for (int j = 0; j < bombnum; j++)
        {
            bomb[i, j].x = -1;
            bomb[i, j].y = -1;
            bomb[i, j].distance = 1;
            bomb[i, j].timesec = -1;
            bomb[i, j].exist = false;
            bomb[i, j].onfire = -1;
            bomb[i, j].bombdirection = 0;
        }

    for (int i = 0; i < ainum; i++)
        for (int j = 0; j < num; j++)
            for (int k = 0; k < num; k++)
            {
                aimap[i, j, k].pass = 0;
                aimap[i, j, k].value = 0;
                aimap[i, j, k].ratvalue = 0;
            }
```

```
            for (int i = 0; i < 2; i++)
            {
                wayindex[i].index = 0;
                wayindex[i].success = 0;
                wayindex[i].direct = 0;
                wayindex[i].back = 0;
                for (int j = 0; j < aiwaymax; j++)
                {
                    aiway[i, j].valuex = 0;
                    aiway[i, j].valuey = 0;
                }
            }
            for (int i = 0; i < ainum; i++)
            {
                aiindex[i] = 0;
                movingflag[i] = false;
            }
            for (int i = 0; i < ainum; i++)
                aidirection[i] = 0;
        }
        private void generate_map()
        {
            //起始=0，通(沙路)=-1，不通(冰牆)=-2，樹=1，玩家起始位置四角=-3 ,炸彈=2
            int[,] x = new int[num, num];

            for (int a = 0; a < num; a++)
                for (int b = 0; b < num; b++)
                {
                    if (a == 0 || a == num - 1 || b == 0 || b == num - 1 || (a == (num - 1) / 2 && b == (num - 1) / 2)) //不通=-2
                        x[a, b] = -2;
                    else if ((a == (num - 1) / 2 - 1 || a == (num - 1) / 2 || a == (num - 1) / 2 + 1) && (b == (num - 1) / 2 - 1 || b == (num - 1) / 2 || b == (num - 1) / 2 + 1)) //通=-1
                        x[a, b] = -1;
                    else
                        x[a, b] = 0; //起始=0
```

```
        }

int tmp, rc, dc, lc, indexx, indexy;
for (int i = 0; i < 4; i++)
{
        if (i == 0 || i == 1)
        {
                if (i == 0)
                {
                        indexx = 1;
                        indexy = 1;
                        rc = (num - 1) / 2 - 2;
                        dc = (num - 1) / 2 - 2;
                }
                else
                {
                        indexx = 8;
                        indexy = 8;
                        rc = (num - 1) / 2 - 2;
                        dc = (num - 1) / 2 - 2;
                }
                while (rc != 0 || dc != 0)
                {
                        tmp = fixRand.Next(0, 4);
                        switch (tmp)
                        {
                                case 0: //right
                                        if (rc > 0)
                                        {
                                                x[indexy, indexx + 1] = -1;
                                                indexx++;
                                                rc--;
                                        }
                                        break;
                                case 1: //down
                                        if (dc > 0)
                                        {
                                                x[indexy + 1, indexx] = -1;
```

```
                    indexy++;
                    dc--;
                }
                break;
            case 2: //left
                if (dc > 0 && rc > 0 && rc != 5)
                {
                    x[indexy, indexx - 1] = -1;
                    indexx--;
                    rc++;
                }
                break;
            case 3: //up
                if (dc > 0 && rc > 0 && dc != 5)
                {
                    x[indexy - 1, indexx] = -1;
                    indexy--;
                    dc++;
                }
                break;
        }
    }
}
else
{
    if (i == 2)
    {
        indexx = 13;
        indexy = 1;
        lc = (num - 1) / 2 - 2;
        dc = (num - 1) / 2 - 2;
    }
    else
    {
        indexx = 6;
        indexy = 8;
        lc = (num - 1) / 2 - 2;
        dc = (num - 1) / 2 - 2;
```

```
}
while (lc != 0 || dc != 0)
{
    tmp = fixRand.Next(0, 4);
    switch (tmp)
    {
        case 0: //left
            if (lc > 0)
            {
                x[indexy, indexx - 1] = -1;
                indexx--;
                lc--;
            }
            break;
        case 1: //down
            if (dc > 0)
            {
                x[indexy + 1, indexx] = -1;
                indexy++;
                dc--;
            }
            break;
        case 2: //right
            if (dc > 0 && lc > 0 && lc != 5)
            {
                x[indexy, indexx + 1] = -1;
                indexx++;
                lc++;
            }
            break;
        case 3: //up
            if (dc > 0 && lc > 0 && dc != 5)
            {
                x[indexy - 1, indexx] = -1;
                indexy--;
                dc++;
            }
            break;
```

```
                }
            }
        }

    }
```

x[1, 1] = -3; x[1, 13] = -3; x[13, 1] = -3; x[13, 13] = -3; x[1, 2] = -3; x[2, 2] = -3; x[13, 2] = -3; x[12, 2] = -3; x[1, 12] = -3; x[2, 12] = -3; x[13, 12] = -3; x[12, 12] = -3;//玩家起始位置四角=-3

```
int count = 0;
for (int i = 1; i < num - 1; i++)
    for (int j = 1; j < num - 1; j++)
        if (x[i, j] == 0)
            count++;
int m;
int n;
for (int i = 0; i < Convert.ToInt64(count / 2); i++)
{
    do
    {
        m = fixRand.Next(1, num - 1);
        n = fixRand.Next(1, num - 1);
    }
    while (x[m, n] != 0);
    x[m, n] = -2;
}

count = 0;
for (int i = 1; i < num - 1; i++)
    for (int j = 1; j < num - 1; j++)
        if (x[i, j] == 0 || x[i, j] == -1)
            count++;

for (int i = 0; i < Convert.ToInt64(count / 5*3); i++)
{
    do
```

```
        {
            m = fixRand.Next(1, num - 1);
            n = fixRand.Next(1, num - 1);
        }
        while (x[m, n] != 0 && x[m, n] != -1);
        x[m, n] = 1;
    }

    for (int i = 1; i < num - 1; i++)
        for (int j = 1; j < num - 1; j++)
            if (x[i, j] == 0)
                x[i, j] = -1;

    x[1, 1] = -1; x[1, 13] = -1; x[13, 1] = -1; x[13, 13] = -1; x[1, 2] = -1; x[2, 2] = -1; x[13, 2] = -1; x[12, 2] = -1; x[1, 12] = -1; x[2, 12] = -1; x[13, 12] = -1; x[12, 12] = -1;//玩家起始位置四角=-1

    int randnum;
    for (int i = 1; i < num - 1; i++)
        for (int j = 1; j < num - 1; j++)
        {
            if (x[i, j] == 1)//樹的地方放特殊道具
            {
                randnum = fixRand.Next(0, 1001);
                if (randnum >= 0 && randnum <= 300)
                    map[i, j].item = 0;//沒有特殊道具
                else if (randnum > 300 && randnum <= 400)
                    map[i, j].item = 1;//炸彈數增加
                else if (randnum > 400 && randnum <= 550)
                    map[i, j].item = 2;//補血
                else if (randnum > 550 && randnum <= 700)
                    map[i, j].item = 3;//推炸彈
                else
                    map[i, j].item = 4;//增加炸彈範圍
            }
        }

    for (int a = 0; a < num; a++)
```

```
                for (int b = 0; b < num; b++)
                    map[a, b].tag = x[a, b];

            }
            public void bombvalue(int playernum, int x, int y, ref int index)
            {
                /*int countexitst = 0 ;
                for(int i=0;i<bombnum;i++)
                    if
                */


                bombtimer.Enabled = true;
                if (map[(y - topy) / mapheight, (x - leftx) / mapwidth].tag != 2)
                {
                    bomb[playernum - 1, index].exist = true;
                    bomb[playernum - 1, index].distance = player[playernum -
1].bombdistance;
                    bomb[playernum - 1, index].x = x;
                    bomb[playernum - 1, index].y = y;
                    bomb[playernum - 1, index].timesec = 30;
                    map[(y - topy) / mapheight, (x - leftx) / mapwidth].bombplayer =
playernum;
                    map[(y - topy) / mapheight, (x - leftx) /
mapwidth].bomblocationindex = index;
                    index = (index + 1) % bombnum;

                    map[(y - topy) / mapheight, (x - leftx) / mapwidth].tag = 2;

                    if (player[playernum - 1].bombcount > 0)
                        player[playernum - 1].bombcount--;

                    uirefresh();

                    Game.Invalidate();
                }
            }
            public void bombsplash(int i, int j, int mapi, int mapj)
```

```
{
    map[mapi, mapj].fire = true;
    map[mapi, mapj].tag = -1;

    map[mapi, mapj].bombplayer = -1;
    map[mapi, mapj].bomblocationindex = -1;

    for (int l = mapj - 1; l >= mapj - bomb[i, j].distance; l--)
    {
        if (map[mapi, l].tag == -2)
            break;
        if (map[mapi, l].tag == 1)
        {
            map[mapi, l].tag = -1;
            map[mapi, l].fire = true;
            break;
        }
        else
        {
            map[mapi, l].tag = -1;
            map[mapi, l].fire = true;
        }
    }


    for (int r = mapj + 1; r <= mapj + bomb[i, j].distance; r++)
    {
        if (map[mapi, r].tag == -2)
            break;
        if (map[mapi, r].tag == 1)
        {
            map[mapi, r].tag = -1;
            map[mapi, r].fire = true;
            break;
        }
        else
        {
            map[mapi, r].tag = -1;
```

```
                map[mapi, r].fire = true;
            }
        }

        for (int u = mapi - 1; u >= mapi - bomb[i, j].distance; u--)
        {
            if (map[u, mapj].tag == -2)
                break;
            if (map[u, mapj].tag == 1)
            {
                map[u, mapj].tag = -1;
                map[u, mapj].fire = true;
                break;
            }
            else
            {
                map[u, mapj].fire = true;
                map[u, mapj].tag = -1;
            }
        }

        for (int d = mapi + 1; d <= mapi + bomb[i, j].distance; d++)
        {
            if (map[d, mapj].tag == -2)
                break;
            if (map[d, mapj].tag == 1)
            {
                map[d, mapj].tag = -1;
                map[d, mapj].fire = true;
                break;
            }
            else
            {
                map[d, mapj].tag = -1;
                map[d, mapj].fire = true;
            }
        }
    }
```

```
public void firetoroad(int i, int j, int mapi, int mapj)
{
    map[mapi, mapj].fire = false;
    map[mapi, mapj].firetoroadflag = 1;

    for (int l = mapj - 1; l >= mapj - bomb[i, j].distance; l--)
    {
        if (map[mapi, l].tag == -2)
            break;
        if (map[mapi, l].fire == true)
        {
            map[mapi, l].fire = false;
            map[mapi, l].firetoroadflag = 1;
        }
    }

    for (int r = mapj + 1; r <= mapj + bomb[i, j].distance; r++)
    {
        if (map[mapi, r].tag == -2)
            break;
        if (map[mapi, r].fire == true)
        {
            map[mapi, r].fire = false;
            map[mapi, r].firetoroadflag = 1;
        }
    }


    for (int u = mapi - 1; u >= mapi - bomb[i, j].distance; u--)
    {
        if (map[u, mapj].tag == -2)
            break;
        if (map[u, mapj].fire == true)
        {
            map[u, mapj].fire = false;
            map[u, mapj].firetoroadflag = 1;
        }
```

```
        }

        for (int d = mapi + 1; d <= mapi + bomb[i, j].distance; d++)
        {
            if (map[d, mapj].tag == -2)
                break;
            if (map[d, mapj].fire == true)
            {
                map[d, mapj].fire = false;
                map[d, mapj].firetoroadflag = 1;
            }

        }
    }

    public void uirefresh()
    {
        player1_life.Text = player[0].life.ToString();
        player2_life.Text = player[1].life.ToString();
        player3_life.Text = player[2].life.ToString();
        player4_life.Text = player[3].life.ToString();
        player1_bomb.Text = player[0].bombcount.ToString();
        player2_bomb.Text = player[1].bombcount.ToString();
        player3_bomb.Text = player[2].bombcount.ToString();
        player4_bomb.Text = player[3].bombcount.ToString();
        if (t % 10 == 0)
        {
            minute.Text = "0" + ((t / 10) / 60).ToString();
            if ((t / 10) % 60 < 10)
                second.Text = "0" + ((t / 10) % 60).ToString();
            else
                second.Text = ((t / 10) % 60).ToString();
        }
    }

    public void player1_specialitem(int playernum, int item)
    {
        switch (item)
```

```
        {
            case 1://增加炸彈數量
                player[playernum - 1].bombcount++;
                uirefresh();
                break;
            case 2://增加血量
                if (player[playernum - 1].life < 5)
                    player[playernum - 1].life++;
                uirefresh();
                break;
            case 3://推炸彈效果 timer 啟動，開始倒數
                player[playernum - 1].pushtime += 50;
                //pushtimer.Enabled = true;
                player1_push.Visible = true;
                break;
            case 4://增加炸彈範圍
                player[playernum - 1].bombdistance += 1;
                break;
        }
    }

    public void player2_specialitem(int playernum, int item)
    {
        switch (item)
        {
            case 1://增加炸彈數量
                player[playernum - 1].bombcount++;
                uirefresh();
                break;
            case 2://增加血量
                player[playernum - 1].life++;
                uirefresh();
                break;
            case 3://推炸彈效果 timer 啟動，開始倒數，每吃到道具延長效果時
間 5 秒
                player[playernum - 1].pushtime += pushtimerload;
                //pushtimer2.Enabled = true;
                player2_push.Visible = true;
```

```
                    break;
                case 4://增加炸彈範圍
                    player[playernum - 1].bombdistance += 1;
                    break;
            }
        }

        public bool iswaymove(int playno, int index, int direction)//上：1；左：2；
下：3；右：4
        {
            bool wayflag = false;
            int mapi = (bomb[playno, index].y - topy) / mapheight;
            int mapj = (bomb[playno, index].x - leftx) / mapwidth;
            //mapi , mapj 為炸彈所在地圖座標
            if (bomb[playno, index].timesec > 0)
            {
                switch (direction)
                {
                    case 1://上
                        if (map[mapi - 1, mapj].tag == -1)
                            wayflag = true;
                        break;
                    case 2://左
                        if (map[mapi, mapj - 1].tag == -1)
                            wayflag = true;
                        break;
                    case 3://下
                        if (map[mapi + 1, mapj].tag == -1)
                            wayflag = true;
                        break;
                    case 4://右
                        if (map[mapi, mapj + 1].tag == -1)
                            wayflag = true;
                        break;
                }
            }
            if (wayflag == false)
                bomb[playno, index].bombdirection = 0;
```

```
            else
            {
                bomb[playno, index].bombdirection = direction;
                map[mapi, mapj].tag = -1;
            }

            return wayflag;//true->有路   false->沒路
        }
        private void AI(int aino)
        {
            aiindex[aino] = 0;

            for (int i = 0; i < num; i++)
                for (int j = 0; j < num; j++)
                    aimap[aino,i,j].pass = 0;

            aimapanalyze(aino, player[playnum+aino].y, player[playnum +
aino].x);//建立 aimap 1 為 電腦玩家可到地方 0->不可到

            bombdanger(aino);//偵測炸彈
            aimapvalue(aino);//寫入 aimap.value 值

            int besti = 0, bestj = 0,comparevalue = -100000;
            for(int i=0;i< num;i++)
                for(int j = 0;j< num;j++)
                    if(aimap[aino,i,j].value> comparevalue)
                    {
                        comparevalue = aimap[aino, i, j].value;
                        besti = i;
                        bestj = j;
                    }

            if (map[player[aino + playnum].y - 1, player[aino + playnum].x].tag != -1
&& map[player[aino + playnum].y + 1, player[aino + playnum].x].tag != -1 &&
map[player[aino + playnum].y, player[aino + playnum].x - 1].tag != -1 &&
map[player[aino + playnum].y, player[aino + playnum].x + 1].tag != -1)//如果四周都被
鎖起來
            {
```

```
                besti = player[aino + playnum].y;
                bestj = player[aino + playnum].x;
            }

            textBox1.Text = " ";
            textBox1.Text = besti.ToString() + ","+bestj.ToString() + "    ";
            movingflag[aino] = true;

            ratwalk(aino, player[aino + playnum].y, player[aino +
playnum].x,besti,bestj);
        }

        public void aimapanalyze(int aino,int i,int j)
        {
            if (map[i, j].tag == -1 || (player[aino + playnum].y == i && player[aino +
playnum].x == j))//通(沙路)
            {
                if ((player[aino + playnum].y == i && player[aino + playnum].x == j)
&& map[i, j].tag != -1)
                    aimap[aino, i, j].pass = 0;
                else
                    aimap[aino, i, j].pass = 1;
                if (aimap[aino, i - 1, j].pass == 0)
                    aimapanalyze(aino, i - 1, j);//上
                if (aimap[aino, i, j - 1].pass == 0)
                    aimapanalyze(aino, i, j - 1);//左
                if (aimap[aino, i + 1, j].pass == 0)
                    aimapanalyze(aino, i + 1, j);//下
                if (aimap[aino, i, j + 1].pass == 0)
                    aimapanalyze(aino, i, j + 1);//右
            }
            else
                return;
            /*
            Console.Write("\n******\n");
            for(int y = 0;y< num;y++)
            {
                for (int k = 0; k < num; k++)
```

```
                    Console.Write(aimap[aino,y,k].pass.ToString()+" ");
                Console.Write("\n");
            }
            */
        }

        public void bombdanger(int aino)
        {
            int bombi, bombj;
            for (int i = 0; i < 4; i++)
                for (int j = 0; j < bombnum; j++)
                    if (bomb[i, j].exist = true && bomb[i, j].timesec > 0)
                    {
                        bombi = ((bomb[i, j].y) - topy) / mapheight;
                        bombj = ((bomb[i, j].x) - leftx) / mapwidth;

                        aimap[aino, bombi, bombj].pass = 2;

                        for (int l = bombj - 1; l >= bombj - bomb[i, j].distance; l--)
                        {
                            if (map[bombi, l].tag == -2 || map[bombi, l].tag == 1)
                                break;
                            else if(aimap[aino, bombi, l].pass == 1)
                                aimap[aino, bombi, l].pass = 2;//有被炸的危險

                        }

                        for (int r = bombj + 1; r <= bombj + bomb[i, j].distance;
r++)
                        {
                            if (map[bombi, r].tag == -2|| map[bombi, r].tag == 1)
                                break;
                            else if(aimap[aino, bombi, r].pass == 1)
                                aimap[aino,bombi,r].pass = 2;//有被炸的危險
                        }

                        for (int u = bombi - 1; u >= bombi - bomb[i, j].distance; u--)
                        {
```

```
                        if (map[u, bombj].tag == -2|| map[u, bombj].tag == 1)
                            break;
                        else if(aimap[aino, u, bombj].pass == 1)
                            aimap[aino, u, bombj].pass = 2;//有被炸的危險
                    }

                    for (int d = bombi + 1; d <= bombi + bomb[i, j].distance;
d++)
                    {
                        if (map[d, bombj].tag == -2|| map[d, bombj].tag == 1)
                            break;
                        else if(aimap[aino, d, bombj].pass == 1)
                            aimap[aino, d, bombj].pass = 2;//有被炸的危險
                    }
                }

            for (int i = 0; i < num; i++)
                for (int j = 0; j < num; j++)
                    if (map[i, j].fire == true && aimap[aino, i, j].pass == 1)
                        aimap[aino, i, j].pass = 2;
            /*
            Console.Write("\n******\n");
            for (int y = 0; y < num; y++)
            {
                for (int k = 0; k < num; k++)
                    Console.Write(aimap[aino, y, k].pass.ToString() + " ");
                Console.Write("\n");
            }
            */
        }

        public void aimapvalue(int aino)
        {
            int ran = fixRand.Next(0, 100);
            int i, j, x, y,aii,aij;
            int value;
            for(i=0;i< num;i++)
            {
```

```
                for (j = 0; j < num; j++)
                {
                    if (i == 0 || j == 0 || i == 14 || j == 14 || aimap[aino, i, j].pass ==
0)
                        value = -100000;
                    else
                    {
                        value = 30000;
                        if (ran <= 30 && t >= warnt && aino==1)
                            value += fixRand.Next(0, 1000);
                        //判斷四周炸彈數
                        if (aimap[aino, i, j].pass == 2)//炸彈炸的到
                            value = value - 50000;
                        //周圍有人可以炸
                        for (x = 0; x < playnum; x++)
                            if (Math.Abs(i - player[x].y) < 2 && Math.Abs(j -
player[x].x) < 2)
                            {
                                value += 30000;
                                if (i == player[x].y)
                                    value += 30000;
                                if (j == player[x].x)
                                    value += 30000;
                            }
                        //周遭可能會被炸到
                        for (x = -1; x <= 1; x++)
                            for (y = -1; y <= 1; y++)
                                if (aimap[aino, i + x, j + y].pass == 2)
                                    value -= 300;

                        //周遭有樹可以炸
                        for (x = -1; x <= 1; x++)
                            for (y = -1; y <= 1; y++)
                                if (map[i + x, j + y].tag == 1)
                                {
                                    value += 10000;
                                    break;
                                }
```

```
                        //周遭為冰牆
                        for (x = -1; x <= 1; x++)
                            for (y = -1; y <= 1; y++)
                                if (map[i + x, j + y].tag == -2)
                                    value -= 100;

                        aij = player[playnum + aino].x;
                        aii = player[playnum + aino].y;

                        int countdanger = 0;
                        for (x = -1; x <= 1; x++)
                            for (y = -1; y <= 1; y++)
                                if (map[i + x, j + y].tag == -2)
                                    countdanger++;
                        if (countdanger == 3)
                            value -= 1000;

                        if (t < warnt)
                        {
                            value -= (warnt - t) / 400 * Math.Abs(i - 7) *
Math.Abs(i - 7);
                            value -= (warnt - t) / 400 * Math.Abs(j - 7) *
Math.Abs(j - 7);
                        }

                        if (aino == 0 && (Math.Abs(i - player[3].y) < 2 || Math.Abs(j
- player[3].x) < 2))
                            value -= 10000;
                        if ((i < 7 && j < 7) || (i > 7 && j > 7))
                            value += 3000;
                        else
                            value += (Convert.ToInt32(Math.Sqrt((aii - i) * (aii - i)
+ (aij - j) * (aij - j)))) * 2;
                    }
                    aimap[aino, i, j].value = value;
                }
            }
            /*
```

```
            Console.Write("\n****** value ******\n");
            for(i=0;i< num;i++)
            {
                for (j = 0; j < num; j++)
                    Console.Write((aimap[aino,i,j].value/1000).ToString()+" ");
                Console.Write("\n");
            }
            */
        }

        public bool ratwalk(int aino,int aiy,int aix,int goaly,int goalx)//老鼠走迷宮
(aix,aiy)目前座標,(goalx,goaly)目的地坐標
        {
            int t = 0,minvalue,direction;//計算次數
            wayindex[aino].index = 0;//index 歸零

            for(int i=0;i<aiwaymax;i++)
            {
                aiway[aino, i].valuex = num;//將座標都設成 15 初始化(方便
DEBUG)
                aiway[aino, i].valuey = num;//將座標都設成 15 初始化(方便
DEBUG)
            }

            for(int i=0;i< num;i++)//建立一開始老鼠走迷宮的 ratvalue
                for(int j=0;j< num;j++)
                {
                    if (map[i, j].tag != -1)//走不過去
                        aimap[aino, i, j].ratvalue = -1;
                    else if (i == aiy && j == aix)//沙路
                        aimap[aino, i, j].ratvalue = 1;
                    else
                        aimap[aino, i, j].ratvalue = 0;
                }

            do
            {
                for(int i=0;i<wayindex[aino].index;i++)
```

```
                {
                    if (aiway[aino, i].valuex == aix && aiway[aino, i].valuey == aiy)
                    {
                        wayindex[aino].index = i;
                        break;
                    }
                }

                aiway[aino, wayindex[aino].index].valuex = aix;
                aiway[aino, wayindex[aino].index].valuey = aiy;

                if ((aix == goalx && aiy == goaly) || t > 200)//判斷是否到達終點
                    break;

                direction = 1;
                minvalue = aimap[aino, aiy - 1, aix].ratvalue;//上
                if ((minvalue == -1 || minvalue > aimap[aino, aiy + 1, aix].ratvalue)
&& aimap[aino, aiy + 1, aix].ratvalue != -1)//下
                {
                    direction = 2;
                    minvalue = aimap[aino, aiy + 1, aix].ratvalue;
                }
                if ((minvalue == -1 || minvalue > aimap[aino, aiy, aix - 1].ratvalue)
&& aimap[aino, aiy, aix - 1].ratvalue != -1)//左
                {
                    direction = 3;
                    minvalue = aimap[aino, aiy, aix - 1].ratvalue;
                }
                if (minvalue == -1 || minvalue > aimap[aino, aiy, aix + 1].ratvalue
&& aimap[aino, aiy, aix + 1].ratvalue != -1)//右
                {
                    direction = 4;
                    minvalue = aimap[aino, aiy, aix + 1].ratvalue;
                }

                if (minvalue == -1)
                    break;
                else
```

```
                {
                    aimap[aino, aiy, aix].ratvalue += 1;
                    switch (direction)
                    {
                        case 1://上
                            aiy -= 1;
                            break;
                        case 2://下
                            aiy += 1;
                            break;
                        case 3://左
                            aix -= 1;
                            break;
                        case 4://右
                            aix += 1;
                            break;
                        default:
                            break;
                    }
                }

                t++;//計算次數
                wayindex[aino].index++;
            }
            while (true);
            /*
            if (aino == 1)
            {
                Console.Write("\n");
                for (int i = 0; i <= wayindex[aino].index; i++)
                    Console.Write(aiway[aino, i].valuey.ToString() + " " +
aiway[aino, i].valuex.ToString() + "\n");
            }
            */
            if (aiy == goaly && aix == goalx)
                return true;
            else
                return false;
```

```
}

public void detectbomb(int aino)
{
    for (int i = 0; i < num; i++)
        for (int j = 0; j < num; j++)
            aimap[aino, i, j].timesec = 30;//初始化

    int mapi, mapj;
    for(int i=0;i< 4;i++)
        for(int j=0;j<bombnum;j++)
        {
            if(bomb[i,j].exist = true && bomb[i,j].timesec>0)
            {
                mapi = (bomb[i, j].y - topy) / mapheight;
                mapj = (bomb[i, j].x - leftx) / mapwidth;
                aimap[aino, mapi, mapj].timesec = bomb[i, j].timesec;
            }
        }
}

public bool setbomb(int aino)
{
    int aiy = player[playnum + aino].y, aix = player[playnum + aino].x;
    bool set = false;

    for (int l = 1; l <= player[playnum + aino].bombdistance; l++)
        if (map[aiy, aix - l].tag == -2)
            break;
        else if (map[aiy, aix - l].tag == 1)
        {
            set = true;
            break;
        }

    for (int r = 1; r <= player[playnum + aino].bombdistance; r++)
        if (map[aiy, aix + r].tag == -2)
            break;
```

```
                else if (map[aiy, aix + r].tag == 1)
                {
                    set = true;
                    break;
                }

            for (int u = 1; u <= player[playnum + aino].bombdistance; u++)
                if (map[aiy - u, aix].tag == -2)
                    break;
                else if (map[aiy - u, aix].tag == 1)
                {
                    set = true;
                    break;
                }
            for (int d = 1; d <= player[playnum + aino].bombdistance; d++)
                if (map[aiy + d, aix].tag == -2)
                    break;
                else if (map[aiy + d, aix].tag == 1)
                {
                    set = true;
                    break;
                }

            return set;
        }

        public void isdied()
        {
            if (gametimer.Enabled == true)
                for (int i = 0; i < 4; i++)
                    if (player[i].life == 0)
                        play[i].Visible = false;
            if (player[0].life == 0 && player[1].life == 0 && (player[2].life != 0 ||
player[3].life != 0))
            {
                gametimer.Enabled = false;
                MessageBox.Show("Lose!!!");
                Main.Location = new Point(0, 0);
```

```
                gameclose();
            }
            else if (player[2].life == 0 && player[3].life == 0)
            {
                if ((player[0].life == 0 && player[1].life != 0) || (player[1].life == 0 &&
player[0].life != 0))
                {
                    int playernumber;
                    if (player[0].life == 0)
                        playernumber = 1;
                    else
                        playernumber = 0;
                    gametimer.Enabled = false;
                    if (InputBox.Show("請輸入名字", "名字：", true, ref name))
                    {
                        if (name != "")
                            fileopen(playernumber, player[playernumber].life,
player[playernumber].bombcount, (1800 - t) / 10, name);
                        Main.Location = new Point(0, 0);
                        //输入成功后的操作
                    }
                    gameclose();
                }
            }
        }

        public void gameclose()
        {
            gametimer.Enabled = false;
            aitimer.Enabled = false;
            uptimer.Enabled = false;
            downtimer.Enabled = false;
            lefttimer.Enabled = false;
            righttimer.Enabled = false;
            uptimer2.Enabled = false;
            downtimer2.Enabled = false;
            lefttimer2.Enabled = false;
            righttimer2.Enabled = false;
```

```
        timer1.Enabled = false;
        timer2.Enabled = false;
        mutekitimer.Enabled = false;
        mutekitimer2.Enabled = false;
        mutekitimer3.Enabled = false;
        mutekitimer4.Enabled = false;
        bombmovetimer.Enabled = false;
        bombtimer.Enabled = false;
        Game.Enabled = false;
        Game.Visible = false;
        Rank.Enabled = false;
        Rank.Visible = false;
        Character.Enabled = false;
        Character.Visible = false;
        Ready.Enabled = false;
        Ready.Visible = false;
        button1.Enabled = true;
        button2.Enabled = true;
        Main.Enabled = true;
        Main.Visible = true;
}
public void fillmap()
{
        switch (warndirection)
        {
                case 1://上
                        warni -= 1;
                        map[warni, warnj].warn = true;
                        map[warni, warnj].tag = -2;
                        if (warnj + warni == 14)
                                warndirection = 2;
                        break;
                case 2://左
                        warnj -= 1;
                        map[warni, warnj].warn = true;
                        map[warni, warnj].tag = -2;
                        if (warnj == warni)
                                warndirection = 3;
```

```
                    break;
                case 3://下
                    warni += 1;
                    map[warni, warnj].warn = true;
                    map[warni, warnj].tag = -2;
                    if (warni - warnj == 11)
                        warndirection = 4;
                    break;
                case 4://右
                    warnj += 1;
                    map[warni, warnj].warn = true;
                    map[warni, warnj].tag = -2;
                    if (warnj == warni)
                        warndirection = 1;
                    break;
            }

            if (map[warni, warnj].warn == true)
            {
                map[warni, warnj].item = 0; //沒有特殊道具
                map[warni, warnj].fire = false;
                map[warni, warnj].firetoroadflag = 0;
                Game.Invalidate();
            }
        }
    }
}
****************************************Form1.cs****************************************
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

```
namespace NBA_BOMB
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            Main.BackgroundImage = new
Bitmap(Properties.Resources.Main_1000);
            Main.Location = new Point(0, 0);
            Main.Enabled = true;
            Main.Visible = true;
            Game.Enabled = false;
            Game.Visible = false;
            Rank.Enabled = false;
            Rank.Visible = false;
            Character.Enabled = false;
            Character.Visible = false;
            Ready.Enabled = false;
            Ready.Visible = false;

            setvalue();

            if (System.Windows.Forms.SystemInformation.TerminalServerSession)
                return;

            System.Reflection.PropertyInfo aProp =
                typeof(System.Windows.Forms.Control).GetProperty(
                    "DoubleBuffered",
                    System.Reflection.BindingFlags.NonPublic |
                    System.Reflection.BindingFlags.Instance);

            aProp.SetValue(Character, true, null);
            aProp.SetValue(Ready, true, null);
            aProp.SetValue(Game, true, null);
            aProp.SetValue(warning, true, null);
        }
```

```
        private void Form1_Load(object sender, EventArgs e)
        {
            this.Size = new Size(1000, 700);
            SetStyle(ControlStyles.UserPaint, true);
            SetStyle(ControlStyles.AllPaintingInWmPaint, true); //  禁止擦除背景.
            SetStyle(ControlStyles.DoubleBuffer, true); //双缓冲
            this.SetStyle(ControlStyles.OptimizedDoubleBuffer |
ControlStyles.ResizeRedraw | ControlStyles.AllPaintingInWmPaint, true);
        }

        private void button1_Click(object sender, EventArgs e)
        {
            /* togame
            GC.Collect();
            setimage();
            generate_map();
            Game.Location = new Point(0, 0);
            Game.Enabled = true;
            Game.Visible = true;
            Main.Enabled = false;
            Main.Visible = false;
            Rank.Enabled = false;
            Rank.Visible = false;
            character.Enabled = false;
            character.Visible = false;
            Ready.Enabled = false;
            Ready.Visible = false;
            button2.Enabled = false;
            //Game.Invalidate();  系統自動重載

            player1.Location = new Point(map[player[0].y, player[0].x].x,
map[player[0].y, player[0].x].y);
            player2.Location = new Point(map[player[1].y, player[1].x].x,
map[player[1].y, player[1].x].y);
            player3.Location = new Point(map[player[2].y, player[2].x].x,
map[player[2].y, player[2].x].y);
```

```
            player4.Location = new Point(map[player[3].y, player[3].x].x,
map[player[3].y, player[3].x].y);

            player1_push.Visible = false;
            player2_push.Visible = false;

            button1.Enabled = false;
            button1.TabStop = false;
            button2.TabStop = false;

            aitimer.Enabled = true;
            gametimer.Enabled = true;
            */
            setvalue();
            Character.Location = new Point(0, 0);
            Character.Enabled = true;
            Character.Visible = true;
            Main.Enabled = false;
            Main.Visible = false;
            Game.Enabled = false;
            Game.Visible = false;
            Rank.Enabled = false;
            Rank.Visible = false;
            Ready.Enabled = false;
            Ready.Visible = false;

            characterset();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Rank.BackgroundImage = new
Bitmap(Properties.Resources.Rank_1000);
            Rank.Location = new Point(0, 0);
            Rank.Enabled = true;
            Rank.Visible = true;
            Main.Enabled = false;
            Main.Visible = false;
```

```
            Game.Enabled = false;
            Game.Visible = false;
            Character.Enabled = false;
            Character.Visible = false;
            Ready.Enabled = false;
            Ready.Visible = false;
            fileread();
        }

        private void Game_Paint(object sender, PaintEventArgs e)
        {
            //     MessageBox.Show("test");
            if (Game.Enabled == true)
            {
                Graphics dr1 = Graphics.FromImage(scr);// 圖層 1
                Graphics dr2 = Graphics.FromImage(scr);// 圖層 1
                Graphics dr3 = Graphics.FromImage(scr);// 圖層 3 火焰
                Graphics dr4 = Graphics.FromImage(scr);// 圖層 4 特殊道具
                Graphics dr5 = Graphics.FromImage(scr);
                //====================== 撿起道具
======================
                if (player1.Location.X == map[player[0].y, player[0].x].x &&
player1.Location.Y == map[player[0].y, player[0].x].y && map[player[0].y,
player[0].x].item > 0 && player[0].life > 0)
                {
                    player1_specialitem(1, map[player[0].y, player[0].x].item);
                    map[player[0].y, player[0].x].item = 0;
                }
                if (player2.Location.X == map[player[1].y, player[1].x].x &&
player2.Location.Y == map[player[1].y, player[1].x].y && map[player[1].y,
player[1].x].item > 0 && player[1].life > 0)
                {
                    player2_specialitem(2, map[player[1].y, player[1].x].item);
                    map[player[1].y, player[1].x].item = 0;
                }
                //====================== End
============================
```

```
                //====================== 人物 1 部分重劃
=================
                if (timer1.Enabled == true || uptimer.Enabled == true ||
downtimer.Enabled == true || righttimer.Enabled == true || lefttimer.Enabled == true)
                {

                    for (int i = 0; i < 3; i++)
                    {
                        for (int j = 0; j < 3; j++)
                        {
                            switch (map[player[0].y + i - 1, player[0].x + j - 1].tag)
                            {
                                case -1:
                                    dr2.DrawImage(pass, map[player[0].y + i -
1, player[0].x + j - 1].x, map[player[0].y + i - 1, player[0].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                case -2:
                                    if (map[player[0].y + i - 1, player[0].x + j -
1].warn == false)
                                        dr2.DrawImage(notpass,
map[player[0].y + i - 1, player[0].x + j - 1].x, map[player[0].y + i - 1, player[0].x + j -
1].y, mapwidth, mapheight);
                                    break;
                                case 1:
                                    dr2.DrawImage(pass, map[player[0].y + i -
1, player[0].x + j - 1].x, map[player[0].y + i - 1, player[0].x + j - 1].y, mapwidth,
mapheight);
                                    dr2.DrawImage(trash, map[player[0].y + i -
1, player[0].x + j - 1].x, map[player[0].y + i - 1, player[0].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                default:
                                    break;
                            }
                        }
                    }
```

```
                }
                //====================== End ======================
                //====================  人物 2 部分重劃 ============
                if (timer2.Enabled == true || uptimer2.Enabled == true ||
downtimer2.Enabled == true || righttimer2.Enabled == true || lefttimer2.Enabled ==
true)
                {
                    for (int i = 0; i < 3; i++)
                    {
                        for (int j = 0; j < 3; j++)
                        {
                            switch (map[player[1].y + i - 1, player[1].x + j - 1].tag)
                            {
                                case -1:
                                    dr2.DrawImage(pass, map[player[1].y + i -
1, player[1].x + j - 1].x, map[player[1].y + i - 1, player[1].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                case -2:
                                    if (map[player[1].y + i - 1, player[1].x + j -
1].warn == false)
                                        dr2.DrawImage(notpass,
map[player[1].y + i - 1, player[1].x + j - 1].x, map[player[1].y + i - 1, player[1].x + j -
1].y, mapwidth, mapheight);
                                    break;
                                case 1:
                                    dr2.DrawImage(pass, map[player[1].y + i -
1, player[1].x + j - 1].x, map[player[1].y + i - 1, player[1].x + j - 1].y, mapwidth,
mapheight);
                                    dr2.DrawImage(trash, map[player[1].y + i -
1, player[1].x + j - 1].x, map[player[1].y + i - 1, player[1].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                default:
                                    break;
                            }
                        }
                    }
```

```
                }
                //====================== End
==============================
                //====================== AI(0)部分重劃
====================
                if (movingflag[0]==true)
                {
                    for (int i = 0; i < 3; i++)
                    {
                        for (int j = 0; j < 3; j++)
                        {
                            switch (map[player[2].y + i - 1, player[2].x + j - 1].tag)
                            {
                                case -1:
                                    dr2.DrawImage(pass, map[player[2].y + i -
1, player[2].x + j - 1].x, map[player[2].y + i - 1, player[2].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                case -2:
                                    if (map[player[2].y + i - 1, player[2].x + j -
1].warn == false)
                                        dr2.DrawImage(notpass,
map[player[2].y + i - 1, player[2].x + j - 1].x, map[player[2].y + i - 1, player[2].x + j -
1].y, mapwidth, mapheight);
                                    break;
                                case 1:
                                    dr2.DrawImage(pass, map[player[2].y + i -
1, player[2].x + j - 1].x, map[player[2].y + i - 1, player[2].x + j - 1].y, mapwidth,
mapheight);
                                    dr2.DrawImage(trash, map[player[2].y + i -
1, player[2].x + j - 1].x, map[player[2].y + i - 1, player[2].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                default:
                                    break;
                            }
                        }
                    }
```

```
                }
                //====================== End
==============================
                //====================== AI(1)部分重劃
====================
                if (movingflag[1] == true)
                {
                    for (int i = 0; i < 3; i++)
                    {
                        for (int j = 0; j < 3; j++)
                        {
                            switch (map[player[3].y + i - 1, player[3].x + j - 1].tag)
                            {
                                case -1:
                                    dr2.DrawImage(pass, map[player[3].y + i -
1, player[3].x + j - 1].x, map[player[3].y + i - 1, player[3].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                case -2:
                                    if (map[player[3].y + i - 1, player[3].x + j -
1].warn == false)
                                        dr2.DrawImage(notpass,
map[player[3].y + i - 1, player[3].x + j - 1].x, map[player[3].y + i - 1, player[3].x + j -
1].y, mapwidth, mapheight);
                                    break;
                                case 1:
                                    dr2.DrawImage(pass, map[player[3].y + i -
1, player[3].x + j - 1].x, map[player[3].y + i - 1, player[3].x + j - 1].y, mapwidth,
mapheight);
                                    dr2.DrawImage(trash, map[player[3].y + i -
1, player[3].x + j - 1].x, map[player[3].y + i - 1, player[3].x + j - 1].y, mapwidth,
mapheight);
                                    break;
                                default:
                                    break;
                            }
                        }
                    }
```

```
                }
                //====================== End
==============================

                if ((timer1.Enabled == false && uptimer.Enabled == false &&
downtimer.Enabled == false && righttimer.Enabled == false && lefttimer.Enabled ==
false) &&
                    (timer2.Enabled == false && uptimer2.Enabled == false &&
downtimer2.Enabled == false && righttimer2.Enabled == false && lefttimer2.Enabled
== false)&&(movingflag[0]==false)&&movingflag[1]==false && t > 1700)
                {
                    for (int i = 0; i < num; i++)
                    {
                        for (int j = 0; j < num; j++)
                        {
                            if (map[i, j].tag == -1)
                                dr1.DrawImage(pass, map[i, j].x, map[i, j].y,
mapwidth, mapheight);
                            else if (map[i, j].tag == -2)
                                dr1.DrawImage(notpass, map[i, j].x, map[i, j].y,
mapwidth, mapheight);
                            else if (map[i, j].tag == 1)
                            {
                                dr1.DrawImage(pass, map[i, j].x, map[i, j].y,
mapwidth, mapheight);
                                dr1.DrawImage(trash, map[i, j].x, map[i, j].y,
mapwidth, mapheight);
                            }

                            if(map[i, j].item >= 1 && map[i, j].tag == -1)
                                dr4.DrawImage(tool[map[i, j].item], map[i, j].x,
map[i, j].y, mapwidth, mapheight);
                        }
                    }
                }

                //=============== 炸彈移動重劃
============================
```

```
                for(int a=0;a<4;a++)
                    for(int b=0;b<bombnum;b++)
                        if(bomb[a,b].bombdirection>0)
                        {
                            btomi = (bomb[a, b].y - topy) / mapheight;
                            btomj = (bomb[a, b].x - leftx) / mapwidth;
                            for (int i = 0; i < 3; i++)
                                for (int j = 0; j < 3; j++)
                                    if(map[btomi + i - 1, btomj + j - 1].tag==-1)
                                        dr2.DrawImage(pass, map[btomi +
i - 1, btomj + j - 1].x, map[btomi + i - 1, btomj + j - 1].y, mapwidth, mapheight);
                        }
                //================ 化火焰 特殊道具 炸彈
====================
                for (int i = 0; i < num; i++)
                    for (int j = 0; j < num; j++)
                    {
                        if (map[i, j].firetoroadflag == 1)
                        {
                            dr1.DrawImage(pass, map[i, j].x, map[i, j].y,
mapwidth, mapheight);
                            map[i, j].firetoroadflag = 0;
                        }
                        if (map[i, j].fire == true)
                            dr1.DrawImage(pass, map[i, j].x, map[i, j].y,
mapwidth, mapheight);
                        if (map[i, j].item >= 1 && map[i, j].tag == -1)
                            dr4.DrawImage(tool[map[i,j].item], map[i, j].x, map[i,
j].y, mapwidth, mapheight);
                        if (map[i, j].fire == true)
                            dr3.DrawImage(fire, map[i, j].x, map[i, j].y, mapwidth,
mapheight);
                    }

                for (int i = 0; i < 4; i++)
                    for (int j = 0; j < bombnum; j++)
                    {
                        if (bomb[i, j].exist == true)
```

```
                    dr3.DrawImage(bomber, bomb[i, j].x, bomb[i, j].y,
mapwidth, mapheight);
                }

            //====================== End ==================
            //====================== 劃線 =================
            for (int i = 0; i < num; i++)
                dr1.DrawLine(myPen, map[i, 0].x, map[i, 0].y, map[i, num - 1].x
+ mapwidth, map[i, 0].y); //橫線
            for (int j = 0; j < num; j++)
                dr1.DrawLine(myPen, map[0, j].x, map[0, j].y, map[0, j].x,
map[num - 1, j].y + mapheight); //直線

            dr1.DrawLine(myPen, map[0, num - 1].x + mapwidth, map[0, 0].y,
map[0, num - 1].x + mapwidth, map[num - 1, 0].y + mapheight); //最右
            dr1.DrawLine(myPen, map[0, 0].x, map[num - 1, 0].y + mapheight,
map[0, num - 1].x + mapwidth, map[num - 1, 0].y + mapheight); //最下
            //====================== End ==================

            //================== 偵測 player1 被燒到 =======
            if (map[player[0].y, player[0].x].fire == true && mutekitimer.Enabled
== false && player[0].life > 0)
            {
                player[0].life--;
                player1_life.Text = player[0].life.ToString();
                if(player[0].life>0)
                    mutekitimer.Enabled = true;
                else
                    isdied();
            }
            //====================== End ==================
            //================== 偵測 player2 被燒到 =======
            if (map[player[1].y, player[1].x].fire == true &&
mutekitimer2.Enabled == false && player[1].life > 0)
            {
                player[1].life--;
                player2_life.Text = player[1].life.ToString();
                if (player[1].life > 0)
```

```
                    mutekitimer2.Enabled = true;
                else
                    isdied();
            }
            //====================== End ==================

            //==================  偵測 player3 (AI 1)  被燒到  =======
            if (map[player[2].y, player[2].x].fire == true &&
mutekitimer3.Enabled == false && player[2].life > 0)
            {
                player[2].life--;
                player3_life.Text = player[2].life.ToString();
                if(player[2].life>0)
                    mutekitimer3.Enabled = true;
                else
                    isdied();
            }
            //====================== End ==================
            //==================  偵測 player4 (AI 2)  被燒到  =======
            if (map[player[3].y, player[3].x].fire == true &&
mutekitimer4.Enabled == false && player[3].life > 0)
            {
                player[3].life--;
                player4_life.Text = player[3].life.ToString();
                if (player[3].life > 0)
                    mutekitimer4.Enabled = true;
                else
                    isdied();
            }
            //====================== End ==================
            if (map[warni, warnj].warn == true && t > 300 &&
gametimer.Enabled == true)
            {
                dr5.DrawImage(warnwall, map[warni, warnj].x, map[warni,
warnj].y, mapwidth, mapheight);
                for (int i = 0; i < 4; i++)
                    if (player[i].x == warnj && player[i].y == warni)
                    {
```

```
                        player[i].life = 0;
                        isdied();
                    }
                }
                e.Graphics.DrawImage(scr, 0, 0);
            }
            //GC.Collect();
        }

        private void uptimer_Tick(object sender, EventArgs e)
        {
            if (player[0].life <= 0)
                uptimer.Enabled = false;
            while (player1.Location.Y < map[player[0].y, player[0].x].y)
                player[0].y--;
            if (player[0].y == 0)
                uptimer.Enabled = false;
            else if (map[player[0].y - 1, player[0].x].tag == -1)
                player1.Location = new Point(player1.Location.X,
player1.Location.Y - step);
            //============================== 撞到炸彈
========================
            else if(player1.Location.Y == map[player[0].y, player[0].x].y &&
map[player[0].y - 1, player[0].x].tag == 2 && player[0].pushtime > 0)//碰到炸彈
            {
                if(iswaymove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1)==true) //炸彈有路可以走 ; (int playno,int index,int
direction) 上：1 ; 左：2 ; 下：3 ; 右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[0].y - 1, player[0].x].bomblocationindex = -1;
                    //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                }
                else
                {
                    timer1.Enabled = true;
                    uptimer.Enabled = false;
```

```
                }
            }
            //============================== End
==============================
            else
            {
                timer1.Enabled = true;
                uptimer.Enabled = false;
            }
        }

        private void downtimer_Tick(object sender, EventArgs e)
        {
            if (player[0].life <= 0)
                downtimer.Enabled = false;
            while (player1.Location.Y > map[player[0].y, player[0].x].y)
                player[0].y++;
            if (player[0].y == 14)
                downtimer.Enabled = false;
            else if (map[player[0].y + 1, player[0].x].tag == -1)
                player1.Location = new Point(player1.Location.X,
player1.Location.Y + step);
            //============================== 撞到炸彈
========================
            else if (player1.Location.Y == map[player[0].y, player[0].x].y &&
map[player[0].y + 1, player[0].x].tag == 2 && player[0].pushtime > 0)//碰到炸彈
            {
                if (iswaymove(0, map[player[0].y + 1,
player[0].x].bomblocationindex, 3) == true) //炸彈有路可以走 ; (int playno,int index,int
direction) 上：1 ; 左：2 ; 下：3 ; 右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[0].y + 1, player[0].x].bomblocationindex = -1;
                    //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                }
                else
                {
```

```
                    timer1.Enabled = true;
                    downtimer.Enabled = false;
                }
            }
            //============================== End
==============================
            else
            {
                timer1.Enabled = true;
                downtimer.Enabled = false;
            }
        }

        private void righttimer_Tick(object sender, EventArgs e)
        {
            if (player[0].life <= 0)
                righttimer.Enabled = false;
            while (player1.Location.X > map[player[0].y, player[0].x].x)
                player[0].x++;
            if (player[0].x == 14)
                righttimer.Enabled = false;
            else if (map[player[0].y, player[0].x + 1].tag == -1)
                player1.Location = new Point(player1.Location.X + step,
player1.Location.Y);
            //============================== 撞到炸彈
========================
            else if (player1.Location.X == map[player[0].y, player[0].x].x &&
map[player[0].y, player[0].x + 1].tag == 2 && player[0].pushtime > 0)//碰到炸彈
            {
                if (iswaymove(0, map[player[0].y, player[0].x +
1].bomblocationindex, 4) == true) //炸彈有路可以走 ；(int playno,int index,int
direction) 上：1； 左：2； 下：3； 右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[0].y, player[0].x + 1].bomblocationindex = -1;
                    //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                }
```

```
                else
                {
                    timer1.Enabled = true;
                    righttimer.Enabled = false;
                }
            }
            //============================== End
==============================
            else
            {
                timer1.Enabled = true;
                righttimer.Enabled = false;
            }
        }

        private void lefttimer_Tick(object sender, EventArgs e)
        {
            if (player[0].life <= 0)
                lefttimer.Enabled = false;
            while (player1.Location.X < map[player[0].y, player[0].x].x)
                player[0].x--;
            if (player[0].x == 0)
                lefttimer.Enabled = false;
            else if (map[player[0].y, player[0].x - 1].tag == -1)
                player1.Location = new Point(player1.Location.X - step,
player1.Location.Y);
            //============================== 撞到炸彈
==========================
            else if (player1.Location.X == map[player[0].y, player[0].x].x &&
map[player[0].y, player[0].x - 1].tag == 2 && player[0].pushtime > 0)//碰到炸彈
            {
                if (iswaymove(0, map[player[0].y, player[0].x -
1].bomblocationindex, 2) == true) //炸彈有路可以走 ；(int playno,int index,int
direction) 上：1；左：2；下：3；右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[0].y, player[0].x - 1].bomblocationindex = -1;
```

```
                        //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                    }
                    else
                    {
                        timer1.Enabled = true;
                        lefttimer.Enabled = false;
                    }
                }
                //============================== End
==============================
                else
                {
                    timer1.Enabled = true;
                    lefttimer.Enabled = false;
                }
            }

            private void timer1_Tick(object sender, EventArgs e)
            {
                if (player1.Location.Y < map[player[0].y, player[0].x].y)
                    player1.Location = new Point(player1.Location.X,
player1.Location.Y + step);
                else if (player1.Location.Y > map[player[0].y, player[0].x].y)
                    player1.Location = new Point(player1.Location.X,
player1.Location.Y - step);
                else if (player1.Location.X > map[player[0].y, player[0].x].x)
                    player1.Location = new Point(player1.Location.X - step,
player1.Location.Y);
                else if (player1.Location.X < map[player[0].y, player[0].x].x)
                    player1.Location = new Point(player1.Location.X + step,
player1.Location.Y);
                else
                    timer1.Enabled = false;
            }

            private void uptimer2_Tick(object sender, EventArgs e)
            {
```

```
                if (player[1].life <= 0)
                    uptimer2.Enabled = false;
                while (player2.Location.Y < map[player[1].y, player[1].x].y)
                    player[1].y--;
                if (player[1].y == 0)
                    uptimer2.Enabled = false;
                else if (map[player[1].y - 1, player[1].x].tag == -1)
                    player2.Location = new Point(player2.Location.X,
player2.Location.Y - step);
                //============================== 撞到炸彈
=========================
                else if (player2.Location.Y == map[player[1].y, player[1].x].y &&
map[player[1].y - 1, player[1].x].tag == 2 && player[1].pushtime > 0)//碰到炸彈
                {
                    if (iswaymove(1, map[player[1].y - 1,
player[1].x].bomblocationindex, 1) == true) //炸彈有路可以走 ; (int playno,int index,int
direction) 上：1 ; 左：2 ; 下：3 ; 右：4
                    {
                        bombmovetimer.Enabled = true;
                        map[player[1].y - 1, player[1].x].bomblocationindex = -1;
                        //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                    }
                    else
                    {
                        timer2.Enabled = true;
                        uptimer2.Enabled = false;
                    }
                }
                //============================== End
==============================
                else
                {
                    timer2.Enabled = true;
                    uptimer2.Enabled = false;
                }
            }
```

```
        private void downtimer2_Tick(object sender, EventArgs e)
        {
            if (player[1].life <= 0)
                downtimer2.Enabled = false;
            while (player2.Location.Y > map[player[1].y, player[1].x].y)
                player[1].y++;
            if (player[1].y == 14)
                downtimer2.Enabled = false;
            else if (map[player[1].y + 1, player[1].x].tag == -1)
                player2.Location = new Point(player2.Location.X,
player2.Location.Y + step);
            //============================== 撞到炸彈
========================
            else if (player2.Location.Y == map[player[1].y, player[1].x].y &&
map[player[1].y + 1, player[1].x].tag == 2 && player[1].pushtime > 0)//碰到炸彈
            {
                if (iswaymove(1, map[player[1].y + 1,
player[1].x].bomblocationindex, 3) == true) //炸彈有路可以走 ; (int playno,int index,int
direction) 上：1 ; 左：2 ; 下：3 ; 右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[1].y + 1, player[1].x].bomblocationindex = -1;
                    //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                }
                else
                {
                    timer2.Enabled = true;
                    downtimer2.Enabled = false;
                }
            }
            //============================== End
==============================
            else
            {
                timer2.Enabled = true;
                downtimer2.Enabled = false;
            }
```

```
        }

        private void righttimer2_Tick(object sender, EventArgs e)
        {
            if (player[1].life <= 0)
                righttimer2.Enabled = false;
            while (player2.Location.X > map[player[1].y, player[1].x].x)
                player[1].x++;
            if (player[1].x == 14)
                righttimer2.Enabled = false;
            else if (map[player[1].y, player[1].x + 1].tag == -1)
                player2.Location = new Point(player2.Location.X + step,
player2.Location.Y);
            //============================== 撞到炸彈
========================
            else if (player2.Location.X == map[player[1].y, player[1].x].x &&
map[player[1].y, player[1].x + 1].tag == 2 && player[1].pushtime > 0)//碰到炸彈
            {
                if (iswaymove(1, map[player[1].y, player[1].x +
1].bomblocationindex, 4) == true) //炸彈有路可以走 ；(int playno,int index,int
direction) 上：1；左：2；下：3；右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[1].y, player[1].x + 1].bomblocationindex = -1;
                    //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                }
                else
                {
                    timer2.Enabled = true;
                    righttimer2.Enabled = false;
                }
            }
            //============================== End
==============================
            else
            {
                timer2.Enabled = true;
```

```
                righttimer2.Enabled = false;
            }
        }

        private void lefttimer2_Tick(object sender, EventArgs e)
        {
            if (player[1].life <= 0)
                lefttimer2.Enabled = false;
            while (player2.Location.X < map[player[1].y, player[1].x].x)
                player[1].x--;
            if (player[1].x == 0)
                lefttimer2.Enabled = false;
            else if (map[player[1].y, player[1].x - 1].tag == -1)
                player2.Location = new Point(player2.Location.X - step,
player2.Location.Y);
            //============================ 撞到炸彈
========================
            else if (player2.Location.X == map[player[1].y, player[1].x].x &&
map[player[1].y, player[1].x - 1].tag == 2 && player[1].pushtime > 0)//碰到炸彈
            {
                if (iswaymove(1, map[player[1].y, player[1].x -
1].bomblocationindex, 2) == true) //炸彈有路可以走 ；(int playno,int index,int
direction) 上：1；左：2；下：3；右：4
                {
                    bombmovetimer.Enabled = true;
                    map[player[1].y, player[1].x - 1].bomblocationindex = -1;
                    //bombmove(0, map[player[0].y - 1,
player[0].x].bomblocationindex, 1);
                }
                else
                {
                    timer2.Enabled = true;
                    lefttimer2.Enabled = false;
                }
            }
            //============================ End
==============================
            else
```

```
            {
                timer2.Enabled = true;
                lefttimer2.Enabled = false;
            }
        }

        private void timer2_Tick(object sender, EventArgs e)
        {
            if (player2.Location.Y < map[player[1].y, player[1].x].y)
                player2.Location = new Point(player2.Location.X,
player2.Location.Y + step);
            else if (player2.Location.Y > map[player[1].y, player[1].x].y)
                player2.Location = new Point(player2.Location.X,
player2.Location.Y - step);
            else if (player2.Location.X > map[player[1].y, player[1].x].x)
                player2.Location = new Point(player2.Location.X - step,
player2.Location.Y);
            else if (player2.Location.X < map[player[1].y, player[1].x].x)
                player2.Location = new Point(player2.Location.X + step,
player2.Location.Y);
            else
                timer2.Enabled = false;
        }

        private void Form1_KeyUp(object sender, KeyEventArgs e)
        {
            if (Game.Enabled == true && player[0].life > 0)
            {
                if (downtimer.Enabled == true && e.KeyCode == Keys.S)
                {
                    timer1.Enabled = true;
                    downtimer.Enabled = false;
                }
                else if (uptimer.Enabled == true && e.KeyCode == Keys.W)
                {
                    timer1.Enabled = true;
                    uptimer.Enabled = false;
                }
```

```
            else if (righttimer.Enabled == true && e.KeyCode == Keys.D)
            {
                timer1.Enabled = true;
                righttimer.Enabled = false;
            }
            else if (lefttimer.Enabled == true && e.KeyCode == Keys.A)
            {
                timer1.Enabled = true;
                lefttimer.Enabled = false;
            }
        }

        if (Game.Enabled == true && player[1].life > 0)
        {
            if (downtimer2.Enabled == true && e.KeyCode == Keys.Down)
            {
                timer2.Enabled = true;
                downtimer2.Enabled = false;
            }
            else if (uptimer2.Enabled == true && e.KeyCode == Keys.Up)
            {
                timer2.Enabled = true;
                uptimer2.Enabled = false;
            }
            else if (righttimer2.Enabled == true && e.KeyCode == Keys.Right)
            {
                timer2.Enabled = true;
                righttimer2.Enabled = false;
            }
            else if (lefttimer2.Enabled == true && e.KeyCode == Keys.Left)
            {
                timer2.Enabled = true;
                lefttimer2.Enabled = false;
            }
        }
    }

    private void Form1_KeyDown(object sender, KeyEventArgs e)
```

```
        {
            if (Game.Enabled == true && uptimer.Enabled == false &&
downtimer.Enabled == false && righttimer.Enabled == false && lefttimer.Enabled ==
false && timer1.Enabled == false && player[0].life > 0)
            {
                // int direction; 上：1 ; 左：2 ; 下：3 ; 右：4
                if (e.KeyCode == Keys.S)
                {
                    downtimer.Enabled = true;
                    player[0].direction = 3;
                }
                else if (e.KeyCode == Keys.W)
                {
                    uptimer.Enabled = true;
                    player[0].direction = 1;
                }
                else if (e.KeyCode == Keys.D)
                {
                    righttimer.Enabled = true;
                    player[0].direction = 4;
                }
                else if (e.KeyCode == Keys.A)
                {
                    lefttimer.Enabled = true;
                    player[0].direction = 2;
                }
                else if (e.KeyCode == Keys.Space && player[0].bombcount > 0)
                    bombvalue(1, player1.Location.X, player1.Location.Y, ref
player[0].bombindex);
            }

            if (Game.Enabled == true && uptimer2.Enabled == false &&
downtimer2.Enabled == false && righttimer2.Enabled == false && lefttimer2.Enabled
== false && timer2.Enabled == false && player[1].life > 0)
            {
                if (e.KeyCode == Keys.Down)
                {
                    downtimer2.Enabled = true;
```

```
                player[1].direction = 3;
            }
            else if (e.KeyCode == Keys.Up)
            {
                uptimer2.Enabled = true;
                player[1].direction = 1;
            }
            else if (e.KeyCode == Keys.Right)
            {
                righttimer2.Enabled = true;
                player[1].direction = 4;
            }
            else if (e.KeyCode == Keys.Left)
            {
                lefttimer2.Enabled = true;
                player[1].direction = 2;
            }
            else if (e.KeyCode == Keys.L && player[1].bombcount > 0)
                bombvalue(2, player2.Location.X, player2.Location.Y, ref
player[1].bombindex);
        }

        //===============Character===============
        if (Character.Enabled == true)
        {
            if (e.KeyCode == Keys.S &&
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) != 3)
            {
                character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 0;

                selectindex1 += 5;
                if (selectindex1 > 9)
                    selectindex1 -= 10;
                if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0,
selectindex1 > 4 ? selectindex1 - 5 : selectindex1].Tag) == 2 ||
```

```
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) == 4)
                    selectindex1 += 5;
                if (selectindex1 > 9)
                    selectindex1 -= 10;

                character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 1;
            }
            else if (e.KeyCode == Keys.W &&
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) != 3)
            {
                character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 0;

                selectindex1 -= 5;
                if (selectindex1 < 0)
                    selectindex1 += 10;
                if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0,
selectindex1 > 4 ? selectindex1 - 5 : selectindex1].Tag) == 2 ||
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) == 4)
                    selectindex1 -= 5;
                if (selectindex1 < 0)
                    selectindex1 += 10;

                character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 1;
            }
            else if (e.KeyCode == Keys.D &&
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) != 3)
            {
                character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 0;

                int flag = 0;
```

```
                    selectindex1 += 1;
                    if (selectindex1 == 5)
                        selectindex1 = 0;
                    else if (selectindex1 == 10)
                        selectindex1 = 5;
                    if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0,
selectindex1 > 4 ? selectindex1 - 5 : selectindex1].Tag) == 2 ||
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) == 4)
                    {
                        selectindex1 += 1;
                        flag = 1;
                    }
                    if (selectindex1 == 5 && flag == 1)
                        selectindex1 = 0;
                    else if (selectindex1 == 10 && flag == 1)
                        selectindex1 = 5;

                    character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 1;
                }
                else if (e.KeyCode == Keys.A &&
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) != 3)
                {
                    character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 0;

                    int flag = 0;
                    selectindex1 -= 1;
                    if (selectindex1 == -1)
                        selectindex1 = 4;
                    else if (selectindex1 == 4)
                        selectindex1 = 9;
                    if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0,
selectindex1 > 4 ? selectindex1 - 5 : selectindex1].Tag) == 2 ||
Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ? selectindex1 -
5 : selectindex1].Tag) == 4)
```

```
                {
                    selectindex1 -= 1;
                    flag = 1;
                }
                if (selectindex1 == -1 && flag == 1)
                    selectindex1 = 4;
                else if (selectindex1 == 4 && flag == 1)
                    selectindex1 = 9;

                character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 1;
            }
            else if (e.KeyCode == Keys.Space)
            {
                if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0,
selectindex1 > 4 ? selectindex1 - 5 : selectindex1].Tag) == 1)
                {
                    character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 3;
                    p1_ready.Text = "Press Space to cancel";
                }
                else if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0,
selectindex1 > 4 ? selectindex1 - 5 : selectindex1].Tag) == 3)
                {
                    character[selectindex1 > 4 ? 1 : 0, selectindex1 > 4 ?
selectindex1 - 5 : selectindex1].Tag = 1;
                    p1_ready.Text = "Press Space to be ready";
                }

            }

            if (e.KeyCode == Keys.Down &&
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) != 4)
            {
                character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 0;
```

```
                    selectindex2 += 5;
                    if (selectindex2 > 9)
                        selectindex2 -= 10;
                    if (Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0,
selectindex2 > 4 ? selectindex2 - 5 : selectindex2].Tag) == 1 ||
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) == 3)
                        selectindex2 += 5;
                    if (selectindex2 > 9)
                        selectindex2 -= 10;

                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 2;
                }
                else if (e.KeyCode == Keys.Up &&
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) != 4)
                {
                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 0;

                    selectindex2 -= 5;
                    if (selectindex2 < 0)
                        selectindex2 += 10;
                    if (Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0,
selectindex2 > 4 ? selectindex2 - 5 : selectindex2].Tag) == 1 ||
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) == 3)
                        selectindex2 -= 5;
                    if (selectindex2 < 0)
                        selectindex2 += 10;

                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 2;
                }
                else if (e.KeyCode == Keys.Right &&
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) != 4)
```

```
                {
                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 0;

                    int flag = 0;
                    selectindex2 += 1;
                    if (selectindex2 == 5)
                        selectindex2 = 0;
                    else if (selectindex2 == 10)
                        selectindex2 = 5;
                    if (Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0,
selectindex2 > 4 ? selectindex2 - 5 : selectindex2].Tag) == 1 ||
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) == 3)
                    {
                        selectindex2 += 1;
                        flag = 1;
                    }
                    if (selectindex2 == 5 && flag == 1)
                        selectindex2 = 0;
                    else if (selectindex2 == 10 && flag == 1)
                        selectindex2 = 5;

                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 2;
                }
                else if (e.KeyCode == Keys.Left &&
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) != 4)
                {
                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 0;

                    int flag = 0;
                    selectindex2 -= 1;
                    if (selectindex2 == -1)
                        selectindex2 = 4;
                    else if (selectindex2 == 4)
```

```
                    selectindex2 = 9;
                if (Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0,
selectindex2 > 4 ? selectindex2 - 5 : selectindex2].Tag) == 1 ||
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) == 3)
                {
                    selectindex2 -= 1;
                    flag = 1;
                }
                if (selectindex2 == -1 && flag == 1)
                    selectindex2 = 4;
                else if (selectindex2 == 4 && flag == 1)
                    selectindex2 = 9;

                character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 2;
            }
            else if (e.KeyCode == Keys.L)
            {
                if (Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0,
selectindex2 > 4 ? selectindex2 - 5 : selectindex2].Tag) == 2)
                {
                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 4;
                    p2_ready.Text = "Press L to cancel";
                }
                else if (Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0,
selectindex2 > 4 ? selectindex2 - 5 : selectindex2].Tag) == 4)
                {
                    character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ?
selectindex2 - 5 : selectindex2].Tag = 2;
                    p2_ready.Text = "Press L to be ready";
                }
            }

            //Character.Invalidate();

        }
```

```
            e.SuppressKeyPress = true; //讓鍵盤 wasd 按下時沒有聲音
        }

        private void bombtimer_Tick(object sender, EventArgs e)
        {
            // bool enableflag = false;
             //bool fireflag = false;

             int checkcount;//計算被爆炸的炸彈個數

             do
             {
                  checkcount = 0;
                  for (int i = 0; i < 4; i++)
                       for (int j = 0; j < bombnum; j++)
                       {
                            if (bomb[i, j].exist == true)
                            {
                                 //enableflag = true;

                                 if (bomb[i, j].timesec > 0)
                                      bomb[i, j].timesec--;
                                 //================================被爆炸的
炸彈==================================
                                 if (map[(bomb[i, j].y - topy) / mapheight, (bomb[i, j].x -
leftx) / mapwidth].fire == true && bomb[i, j].timesec > 0)
                                 {
                                      bomb[i, j].timesec = 0;
                                      checkcount++;
                                 }
                                 //====================================爆炸
========================================
                                 if (bomb[i, j].timesec == 0 &&
bomb[i,j].bombdirection<=0)
                                 {
                                     // MessageBox.Show("bomb");
                                      //map[(bomb[i, j].y - topy) / mapheight, (bomb[i,
j].x - leftx) / mapwidth].tag = -1;
```

```
                                    bombsplash(i, j, (bomb[i, j].y - topy) / mapheight,
(bomb[i, j].x - leftx) / mapwidth);//爆炸
                                    bomb[i, j].onfire = 10; //火焰持續 0.5 秒
                                    bomb[i, j].timesec = -1;
                                    bomb[i, j].exist = false;

                                    //if(player[i].bombcount<bombattacknum)
                                    player[i].bombcount++;

                                    uirefresh();
                                }
                            }
                        }
                    }
                    while (checkcount>0);

                    for (int i = 0; i < 4; i++)
                        for (int j = 0; j < bombnum; j++)
                        {
                            if (bomb[i, j].onfire >= 0)
                            {
                                //fireflag = true;

                                if (bomb[i, j].onfire > 0)
                                    bomb[i, j].onfire--;

                                if (bomb[i, j].onfire == 0)
                                {
                                    firetoroad(i, j, (bomb[i, j].y - topy) / mapheight,
(bomb[i, j].x - leftx) / mapwidth);
                                    bomb[i, j].onfire = -1;
                                    bomb[i, j].x = -1;
                                    bomb[i, j].y = -1;
                                }
                            }
                        }

                    Game.Invalidate();
```

```
        //if (enableflag == false && fireflag == false)
          //      bombtimer.Enabled = false;
}

private void mutekitimer_Tick(object sender, EventArgs e)
{
        player[0].mutekitime--;
        if(player[0].mutekitime == 0)
        {
                player[0].mutekitime = mutekitimerload;
                mutekitimer.Enabled = false;
        }
        if (player1.Visible == true)
                player1.Visible = false;
        else
                player1.Visible = true;
}

private void mutekitimer2_Tick(object sender, EventArgs e)
{
        player[1].mutekitime--;
        if (player[1].mutekitime == 0)
        {
                player[1].mutekitime = mutekitimerload;
                mutekitimer2.Enabled = false;
        }
        if (player2.Visible == true)
                player2.Visible = false;
        else
                player2.Visible = true;
}

private void mutekitimer3_Tick(object sender, EventArgs e)
{
        player[2].mutekitime--;
        if (player[2].mutekitime == 0)
        {
```

```
            player[2].mutekitime = mutekitimerload;
            mutekitimer3.Enabled = false;
        }
        if (player3.Visible == true)
            player3.Visible = false;
        else
            player3.Visible = true;
}

private void mutekitimer4_Tick(object sender, EventArgs e)
{
        player[3].mutekitime--;
        if (player[3].mutekitime == 0)
        {
            player[3].mutekitime = mutekitimerload;
            mutekitimer4.Enabled = false;
        }
        if (player4.Visible == true)
            player4.Visible = false;
        else
            player4.Visible = true;
}

private void bombmovetimer_Tick(object sender, EventArgs e)
{
        int countbombmove = 0;
        int mapi;
        int mapj;
        for (int i = 0; i < 4; i++)
        {
            for (int j = 0; j < bombnum; j++)
            {
                if (bomb[i, j].bombdirection > 0)
                {
                    countbombmove++;
                    mapi = (bomb[i, j].y - topy) / mapheight;
                    mapj = (bomb[i, j].x - leftx) / mapwidth;
                    switch (bomb[i, j].bombdirection)
```

```
                {
                    case 1://上
                        if ((bomb[i, j].y - topy) % mapheight == 0)
                        {
                            if (map[(mapi - 1) < 0 ? 0 : mapi - 1,
mapj].tag != -1 || bomb[i, j].timesec == 0)
                            {
                                bomb[i, j].bombdirection = 0;
                                map[mapi, mapj].tag = 2;
                                map[mapi, mapj].bomblocationindex = j;
                                break;
                            }
                        }
                        bomb[i, j].y -= step;
                        break;
                    case 2://左
                        if ((bomb[i, j].x - leftx) % mapwidth == 0)
                        {
                            if (map[mapi, (mapj - 1) < 0 ? 0 : mapj -
1].tag != -1 || bomb[i, j].timesec == 0)
                            {
                                bomb[i, j].bombdirection = 0;
                                map[mapi, mapj].tag = 2;
                                map[mapi, mapj].bomblocationindex = j;
                                break;
                            }
                        }
                        bomb[i, j].x -= step;
                        break;
                    case 3://下
                        if ((bomb[i, j].y - topy) % mapheight == 0)
                        {
                            if (map[(mapi + 1) > 14 ? 14 : mapi + 1,
mapj].tag != -1 || bomb[i, j].timesec == 0)
                            {
                                bomb[i, j].bombdirection = 0;
                                map[mapi, mapj].tag = 2;
                                map[mapi, mapj].bomblocationindex = j;
```

```
                                        break;
                                    }
                                }
                                bomb[i, j].y += step;
                                break;
                            case 4://右
                                if ((bomb[i, j].x - leftx) % mapwidth == 0)
                                {
                                    if (map[mapi, (mapj + 1) > 14 ? 14 : mapj +
1].tag != -1 || bomb[i, j].timesec == 0)
                                    {
                                        bomb[i, j].bombdirection = 0;
                                        map[mapi, mapj].tag = 2;
                                        map[mapi, mapj].bomblocationindex = j;
                                        break;
                                    }
                                }
                                bomb[i, j].x += step;
                                break;
                        }
                    }
                }
            }

            Game.Invalidate();

            if (countbombmove == 0)
                bombmovetimer.Enabled = false;

        }

        private void aitimer_Tick(object sender, EventArgs e)
        {
            //右上角 ai 移動
            if (player[playnum].life > 0)
            {
                if (movingflag[0] == false)
                    AI(0);//呼叫右上角 ai
```

```
                else
                {
                    detectbomb(0);
                    if ((player3.Location.X - leftx) % mapwidth == 0 &&
(player3.Location.Y - topy) % mapheight == 0)
                    {
                        if (aiindex[0] == wayindex[0].index)
                        {
                            movingflag[0] = false;
                            aiindex[0] = 0;
                            if (player[2].bombcount > 0)
                                if (setbomb(0) == true)
                                    bombvalue(3, player3.Location.X,
player3.Location.Y, ref player[2].bombindex);
                        }
                        else if (player[playnum].y == aiway[0, aiindex[0]].valuey
&& player[playnum].x == aiway[0, aiindex[0]].valuex)
                            aiindex[0]++;
                        if ((Math.Abs(player[2].y - aiway[0, aiindex[0]].valuey) +
Math.Abs(player[2].x - aiway[0, aiindex[0]].valuex)) >= 2)
                            movingflag[0] = false;
                        if (aimap[0, aiway[0, aiindex[0]].valuey, aiway[0,
aiindex[0]].valuex].timesec < 20)
                            movingflag[0] = false;
                        if (map[aiway[0, aiindex[0]].valuey, aiway[0,
aiindex[0]].valuex].fire == true)
                            movingflag[0] = false;
                        if (map[aiway[0, aiindex[0]].valuey, aiway[0,
aiindex[0]].valuex].tag != -1)
                            movingflag[0] = false;
                    }

                    if (movingflag[0] == true)
                    {
                        aidirection[0] = 0;
                        if (player3.Location.Y > aiway[0, aiindex[0]].valuey *
mapheight + topy)
                            aidirection[0] = 1;//上
```

```
                else if (player3.Location.X > aiway[0, aiindex[0]].valuex *
mapwidth + leftx)
                    aidirection[0] = 2;//左
                else if (player3.Location.Y < aiway[0, aiindex[0]].valuey *
mapheight + topy)
                    aidirection[0] = 3;//下
                else if (player3.Location.X < aiway[0, aiindex[0]].valuex *
mapwidth + leftx)
                    aidirection[0] = 4;//右
                switch (aidirection[0])
                {
                    case 1:
                        player3.Location = new
Point(player3.Location.X, player3.Location.Y - aistep);
                        player[2].y = (player3.Location.Y - topy) /
mapheight;
                        break;
                    case 2:
                        player3.Location = new Point(player3.Location.X
- aistep, player3.Location.Y);
                        player[2].x = (player3.Location.X - leftx) /
mapwidth;
                        break;
                    case 3:
                        player3.Location = new
Point(player3.Location.X, player3.Location.Y + aistep);
                        player[2].y = (player3.Location.Y - topy) /
mapheight;
                        break;
                    case 4:
                        player3.Location = new Point(player3.Location.X
+ aistep, player3.Location.Y);
                        player[2].x = (player3.Location.X - leftx) /
mapwidth;
                        break;
                    default:
                        movingflag[0] = false;
                        break;
```

```
                    }
                }

            }
        }

        if (player[playnum + 1].life > 0)
        {
            if (movingflag[1] == false)
                AI(1);//呼叫右上角 ai
            else
            {
                detectbomb(1);
                if (player[playnum + 1].y == aiway[1, aiindex[1]].valuey &&
player[playnum + 1].x == aiway[1, aiindex[1]].valuex && (player4.Location.X - leftx) %
mapwidth == 0 && (player4.Location.Y - topy) % mapheight == 0)
                {
                    if (aiindex[1] == wayindex[1].index)
                    {
                        movingflag[1] = false;
                        aiindex[1] = 0;
                        if (player[3].bombcount > 0)
                            bombvalue(4, player4.Location.X,
player4.Location.Y, ref player[3].bombindex);
                    }
                    else
                        aiindex[1]++;
                }
                else if ((player4.Location.Y - topy) % mapheight != 0 ||
(player4.Location.X - leftx) % mapwidth != 0 || (map[aiway[1, aiindex[1]].valuey,
aiway[1, aiindex[1]].valuex].fire == false && aimap[1, aiway[1, aiindex[1]].valuey,
aiway[1, aiindex[1]].valuex].timesec > 20))
                {
                    //Console.Write(aiway[0, aiindex[0]].valuey.ToString() +
"\n");
                    if ((aiway[1, aiindex[1]].valuey * mapheight + topy) <
player4.Location.Y && map[aiway[1, aiindex[1]].valuey, aiway[1,
aiindex[1]].valuex].tag == -1)
```

```
                {//上
                    player4.Location = new Point(player4.Location.X,
player4.Location.Y - aistep);
                    player[3].y = (player4.Location.Y - topy) / mapheight;
                }
                else if ((aiway[1, aiindex[1]].valuey * mapheight + topy) >
player4.Location.Y && map[aiway[1, aiindex[1]].valuey, aiway[1,
aiindex[1]].valuex].tag == -1)
                {//下
                    player4.Location = new Point(player4.Location.X,
player4.Location.Y + aistep);
                    player[3].y = (player4.Location.Y - topy) / mapheight;
                    // if ((player3.Location.Y - topy) % mapheight != 0)
                    //    player[2].y += 1;
                }
                else if ((aiway[1, aiindex[1]].valuex * mapwidth + leftx) <
player4.Location.X && map[aiway[1, aiindex[1]].valuey, aiway[1,
aiindex[1]].valuex].tag == -1)
                {
                    player4.Location = new Point(player4.Location.X -
aistep, player4.Location.Y);
                    player[3].x = (player4.Location.X - leftx) / mapwidth;
                }
                else if ((aiway[1, aiindex[1]].valuex * mapwidth + leftx) >
player4.Location.X && map[aiway[1, aiindex[1]].valuey, aiway[1,
aiindex[1]].valuex].tag == -1)
                {
                    player4.Location = new Point(player4.Location.X +
aistep, player4.Location.Y);
                    player[3].x = (player4.Location.X - leftx) / mapwidth;
                    // if ((player3.Location.X - leftx) % mapwidth != 0)
                    //    player[2].x += 1;
                }
                else if (aiindex[1] < wayindex[1].index)
                    aiindex[1]++;
                // else
                    // movingflag[1] = false;
            }
```

```
            }
        }
    }

    private void gametimer_Tick(object sender, EventArgs e)
    {
        if (player[0].pushtime > 0)
            player[0].pushtime--;
        if (player[0].pushtime == 0)
            player1_push.Visible = false;
        if (player[1].pushtime > 0)
            player[1].pushtime--;
        if (player[1].pushtime == 0)
            player2_push.Visible = false;
        if (t < warnt)
        {
            warindex = (warindex + 1) % 6;
            warning.Invalidate();
            if (t % 10 == 0 && t > 300)
                fillmap();
        }
        if (t > 0)
        {
            t--;
            if (t % 10 == 0)
                uirefresh();
        }
        else
        {
            gametimer.Enabled = false;
            MessageBox.Show("Lose!!!");
            Main.Location = new Point(0, 0);
            gameclose();
        }
    }

    private void warning_Paint(object sender, PaintEventArgs e)
```

```
        {
            if (t < warnt)
            {
                Graphics dr1 = Graphics.FromImage(scr1);// 圖層 1
                dr1.DrawImage(war[warindex], 0, 0,warnwidth,warnheight);
                e.Graphics.DrawImage(scr1, 0, 0);
            }
        }

        private void readytimer_Tick(object sender, EventArgs e)
        {
            readyload--;
            readytime.Text = readyload.ToString();

            if(readyload == 0)
            {
                GC.Collect();
                setimage();
                generate_map();
                Game.Location = new Point(0, 0);
                Game.Enabled = true;
                Game.Visible = true;
                Main.Enabled = false;
                Main.Visible = false;
                Rank.Enabled = false;
                Rank.Visible = false;
                Character.Enabled = false;
                Character.Visible = false;
                Ready.Enabled = false;
                Ready.Visible = false;
                button2.Enabled = false;
                //Game.Invalidate(); 系統自動重載

                player1.Location = new Point(map[player[0].y, player[0].x].x,
map[player[0].y, player[0].x].y);
                player2.Location = new Point(map[player[1].y, player[1].x].x,
map[player[1].y, player[1].x].y);
```

```
                    player3.Location = new Point(map[player[2].y, player[2].x].x,
map[player[2].y, player[2].x].y);
                    player4.Location = new Point(map[player[3].y, player[3].x].x,
map[player[3].y, player[3].x].y);

                    player1_push.Visible = false;
                    player2_push.Visible = false;

                    button1.Enabled = false;
                    button1.TabStop = false;
                    button2.TabStop = false;

                    aitimer.Enabled = true;
                    gametimer.Enabled = true;
                    //setvalue();
                }
            }

            private void button3_Click(object sender, EventArgs e)
            {
                //Main.BackgroundImage = new
Bitmap(Properties.Resources.Main_1000);
                Main.Location = new Point(0, 0);
                Main.Enabled = true;
                Main.Visible = true;
                Game.Enabled = false;
                Game.Visible = false;
                Rank.Enabled = false;
                Rank.Visible = false;
                Character.Enabled = false;
                Character.Visible = false;
                Ready.Enabled = false;
                Ready.Visible = false;
            }
        }
}
********** Character_function.cs**********
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace NBA_BOMB
{
    public partial class Form1 : Form
    {
        PictureBox[,] character = new PictureBox[2, 5];
        Image[,] nba = new Image[2, 5];
        Image[,] nbashow = new Image[2, 5];
        Image[,] nbaplayer = new Image[2, 5];
        String[,] nbaname = new String[2, 5];
        const int characterx = 320;
        const int charactery = 500;
        int selectindex1 = -1;
        int selectindex2 = -1;
        int flash = 1;


        private void characterset()
        {
            Character.BackgroundImage = new
Bitmap(Properties.Resources.Character_1000);

            character[0, 0] = charactera1; character[0, 1] = charactera2;
character[0, 2] = charactera3; character[0, 3] = charactera4; character[0, 4] =
charactera5;
            character[1, 0] = characterb1; character[1, 1] = characterb2;
character[1, 2] = characterb3; character[1, 3] = characterb4; character[1, 4] =
characterb5;
```

```
            nba[0, 0] = new Bitmap(Properties.Resources.Jeremy_60); nba[0, 1] =
new Bitmap(Properties.Resources.Michael_60); nba[0, 2] = new
Bitmap(Properties.Resources.LeBron_60); nba[0, 3] = new
Bitmap(Properties.Resources.Wade_60); nba[0, 4] = new
Bitmap(Properties.Resources.Iverson_60);
            nba[1, 0] = new Bitmap(Properties.Resources.McGee_60); nba[1, 1] =
new Bitmap(Properties.Resources.Kobe_60); nba[1, 2] = new
Bitmap(Properties.Resources.Curry_60); nba[1, 3] = new
Bitmap(Properties.Resources.Duncan_60); nba[1, 4] = new
Bitmap(Properties.Resources.Yao_60);

            nbashow[0, 0] = new Bitmap(Properties.Resources.Jeremy_275x275);
nbashow[0, 1] = new Bitmap(Properties.Resources.Michael_300x250); nbashow[0,
2] = new Bitmap(Properties.Resources.LeBron_250x300); nbashow[0, 3] = new
Bitmap(Properties.Resources.Wade_275x275); nbashow[0, 4] = new
Bitmap(Properties.Resources.Iverson_300x250);
            nbashow[1, 0] = new Bitmap(Properties.Resources.McGee_300x250);
nbashow[1, 1] = new Bitmap(Properties.Resources.Kobe_300x250); nbashow[1, 2] =
new Bitmap(Properties.Resources.Curry_275x275); nbashow[1, 3] = new
Bitmap(Properties.Resources.Duncan_300x250); nbashow[1, 4] = new
Bitmap(Properties.Resources.Yao_275x275);

            nbaplayer[0, 0] = new Bitmap(Properties.Resources.Jeremy_40);
nbaplayer[0, 1] = new Bitmap(Properties.Resources.Michael_40); nbaplayer[0, 2] =
new Bitmap(Properties.Resources.LeBron_40); nbaplayer[0, 3] = new
Bitmap(Properties.Resources.Wade_40); nbaplayer[0, 4] = new
Bitmap(Properties.Resources.Iverson_40);
            nbaplayer[1, 0] = new Bitmap(Properties.Resources.McGee_40);
nbaplayer[1, 1] = new Bitmap(Properties.Resources.Kobe_40); nbaplayer[1, 2] = new
Bitmap(Properties.Resources.Curry_40); nbaplayer[1, 3] = new
Bitmap(Properties.Resources.Duncan_40); nbaplayer[1, 4] = new
Bitmap(Properties.Resources.Yao_40);

            nbaname[0, 0] = "Jeremy Lin"; nbaname[0, 1] = "Michael Jordan";
nbaname[0, 2] = "LeBron James"; nbaname[0, 3] = "Dwyane Wade"; nbaname[0, 4]
= "Allen Iverson";
```

```
            nbaname[1, 0] = "JaVale McGee"; nbaname[1, 1] = "Kobe Bryant";
nbaname[1, 2] = "Stephen Curry"; nbaname[1, 3] = "Tim Duncan"; nbaname[1, 4] =
"Yao Ming";

            p1_label.Location = new Point(50, 20);
            p2_label.Location = new Point(685, 20);
            verus.Location = new Point(368, 20);
            p1_character.Location = new Point(40, 100);
            p2_character.Location = new Point(660, 100);
            p1_name.Location = new Point(60, 400);
            p2_name.Location = new Point(675, 400);
            p1_howtoplay.Location = new Point(95, 470);
            p1_howtoplay.Image = new
Bitmap(Properties.Resources.howtoplayp1);
            p2_howtoplay.Location = new Point(720, 470);
            p2_howtoplay.Image = new
Bitmap(Properties.Resources.howtoplayp2);
            p1_ready.Location = new Point(40, 565);
            p1_ready.Text = "Press Space to be ready";
            p2_ready.Location = new Point(680, 565);
            p2_ready.Text = "Press L to be ready";


            for (int i = 0; i < 2; i++)
                for (int j = 0; j < 5; j++)
                {
                    character[i, j].BackColor = Color.LightBlue;
                    character[i, j].Image = nba[i, j];
                    character[i, j].Tag = 0;
                    character[i, j].Location = new Point(characterx + j * 70,
charactery + i * 70);
                }

            character[0, 2].Tag = 1;
            character[1, 2].Tag = 2;

            character[0, 2].Invalidate();
            character[1, 2].Invalidate();
```

```
            selectindex1 = 2; //第一列中間
            selectindex2 = 7; //第二列中間

            selecttimer.Enabled = true;

        }

        private void selecttimer_Tick(object sender, EventArgs e)
        {
            if (Convert.ToInt64(character[selectindex1 > 4 ? 1 : 0, selectindex1 >
4 ? selectindex1 - 5 : selectindex1].Tag) == 3 &&
Convert.ToInt64(character[selectindex2 > 4 ? 1 : 0, selectindex2 > 4 ? selectindex2 -
5 : selectindex2].Tag) == 4)
            {
                selecttimer.Enabled = false;
                p1_start.Size = p1_character.Size - new Size(50, 50); ;
                p1_start.Image = nbashow[selectindex1 > 4 ? 1 : 0, selectindex1 >
4 ? selectindex1 - 5 : selectindex1];
                p2_start.Size = p2_character.Size - new Size(50, 50); ;
                p2_start.Image = nbashow[selectindex2 > 4 ? 1 : 0, selectindex2 >
4 ? selectindex2 - 5 : selectindex2];

                player1.Image = nbaplayer[selectindex1 > 4 ? 1 : 0, selectindex1 >
4 ? selectindex1 - 5 : selectindex1]; //輸入 player1 圖片
                player2.Image = nbaplayer[selectindex2 > 4 ? 1 : 0, selectindex2 >
4 ? selectindex2 - 5 : selectindex2]; //輸入 player2 圖片
                //==========隨機選擇電腦玩家圖片==========
                int p3, p4;
                do
                {
                    p3 = fixRand.Next(0, 10);
                    if (p3 != selectindex1 && p3 != selectindex2)
                    {
                        switch (p3)
                        {
                            case 0:
                            case 3:
```

```
                    case 7:
                    case 9:
                        p3_start.Size = new Size(275, 275);
                        break;
                    case 1:
                    case 4:
                    case 5:
                    case 6:
                    case 8:
                        p3_start.Size = new Size(300, 250);
                        break;
                    case 2:
                        p3_start.Size = new Size(250, 300);
                        break;
                }
                p3_start.Size =    new Size(225, 225);
                p3_start.Image = nbashow[p3 > 4 ? 1 : 0, p3 > 4 ? p3 - 5 :
p3];
                player3.Image = nbaplayer[p3 > 4 ? 1 : 0, p3 > 4 ? p3 - 5 :
p3];
                break;
            }
        }
        while (true);

        do
        {
            p4 = fixRand.Next(0, 10);
            if (p4 != selectindex1 && p4 != selectindex2 && p4 != p3)
            {
                switch(p4)
                {
                    case 0 :
                    case 3 :
                    case 7 :
                    case 9 :
                        p4_start.Size = new Size(275, 275);
                        break;
```

```
                        case 1 :
                        case 4 :
                        case 5 :
                        case 6 :
                        case 8 :
                            p4_start.Size = new Size(300, 250);
                            break;
                        case 2 :
                            p4_start.Size = new Size(250, 300);
                            break;
                    }
                    p4_start.Size = new Size(225, 225);
                    p4_start.Image = nbashow[p4 > 4 ? 1 : 0, p4 > 4 ? p4 - 5 :
p4];
                    player4.Image = nbaplayer[p4 > 4 ? 1 : 0, p4 > 4 ? p4 - 5 :
p4];
                    break;
                }
            }
            while (true);

            //==========切换到 Ready panel==========
            Ready.Location = new Point(0, 0);
            Ready.Enabled = true;
            Ready.Visible = true;
            Main.Enabled = false;
            Main.Visible = false;
            Game.Enabled = false;
            Game.Visible = false;
            Rank.Enabled = false;
            Rank.Visible = false;
            Character.Enabled = false;
            Character.Visible = false;
            readyset();
        }


        if (flash == 1)
```

```
                flash = 0;
            else if (flash == 0)
                flash = 1;

            Character.Invalidate();
        }

        // Tag 0:沒事 ；1 : p1 正在選 ；2 : p2 正在選 ；3 : p1 已選 ；4 : p2 已選
        private void Character_Paint(object sender, PaintEventArgs e)
        {
            for (int i = 0; i < 2; i++)
                for (int j = 0; j < 5; j++)
                {
                    character[i, j].Invalidate();
                }
        }

        private void charactera3_Paint(object sender, PaintEventArgs e)
//LeBron_250x300
        {
            PictureBox pb = (PictureBox)sender;
            if ((Convert.ToInt64(pb.Tag) == 1 && flash == 1) ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(250, 300);
                p1_character.Image = nbashow[0, 2];
                p1_name.Text = nbaname[0, 2];
            }
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(250, 300);
                p2_character.Image = nbashow[0, 2];
                p2_name.Text = nbaname[0, 2];
```

```
            }
        }

        private void characterb3_Paint(object sender, PaintEventArgs e)
//Curry_275x275
        {
            PictureBox pb = (PictureBox)sender;
            if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(275, 275);
                p1_character.Image = nbashow[1, 2];
                p1_name.Text = nbaname[1, 2];
            }
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(275, 275);
                p2_character.Image = nbashow[1, 2];
                p2_name.Text = nbaname[1, 2];
            }
        }

        private void charactera1_Paint(object sender, PaintEventArgs e)
//Jeremy_275x275
        {
            PictureBox pb = (PictureBox)sender;
            if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(275, 275);
                p1_character.Image = nbashow[0, 0];
```

```
                p1_name.Text = nbaname[0, 0];
            }
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(275, 275);
                p2_character.Image = nbashow[0, 0];
                p2_name.Text = nbaname[0, 0];
            }
        }

        private void charactera2_Paint(object sender, PaintEventArgs e)
//Michael_300x250
        {
            PictureBox pb = (PictureBox)sender;
            if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(300, 250);
                p1_character.Image = nbashow[0, 1];
                p1_name.Text = nbaname[0, 1];
            }
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(300, 250);
                p2_character.Image = nbashow[0, 1];
                p2_name.Text = nbaname[0, 1];
            }
        }
```

```
        private void charactera4_Paint(object sender, PaintEventArgs e)
//Wade_275x275
        {
            PictureBox pb = (PictureBox)sender;
            if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(275, 275);
                p1_character.Image = nbashow[0, 3];
                p1_name.Text = nbaname[0, 3];
            }
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(275, 275);
                p2_character.Image = nbashow[0, 3];
                p2_name.Text = nbaname[0, 3];
            }
        }

        private void charactera5_Paint(object sender, PaintEventArgs e)
//Iverson_300x250
        {
            PictureBox pb = (PictureBox)sender;
            if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(300, 250);
                p1_character.Image = nbashow[0, 4];
                p1_name.Text = nbaname[0, 4];
            }
```

```
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(300, 250);
                p2_character.Image = nbashow[0, 4];
                p2_name.Text = nbaname[0, 4];
            }
        }

        private void characterb1_Paint(object sender, PaintEventArgs e)
//McGee_300x250
        {
            PictureBox pb = (PictureBox)sender;
            if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                p1_character.Size = new Size(300, 250);
                p1_character.Image = nbashow[1, 0];
                p1_name.Text = nbaname[1, 0];
            }
            else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
            {
                e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                p2_character.Size = new Size(300, 250);
                p2_character.Image = nbashow[1, 0];
                p2_name.Text = nbaname[1, 0];
            }
        }

        private void characterb2_Paint(object sender, PaintEventArgs e)
//Kobe_300x250
        {
```

```
                PictureBox pb = (PictureBox)sender;
                if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
                {
                    e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                    p1_character.Size = new Size(300, 250);
                    p1_character.Image = nbashow[1, 1];
                    p1_name.Text = nbaname[1, 1];
                }
                else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
                {
                    e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                    p2_character.Size = new Size(300, 250);
                    p2_character.Image = nbashow[1, 1];
                    p2_name.Text = nbaname[1, 1];
                }
            }

            private void characterb4_Paint(object sender, PaintEventArgs e)
//Duncan_300x250
            {
                PictureBox pb = (PictureBox)sender;
                if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
                {
                    e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                    p1_character.Size = new Size(300, 250);
                    p1_character.Image = nbashow[1, 3];
                    p1_name.Text = nbaname[1, 3];
                }
                else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
                {
```

```
                    e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                    p2_character.Size = new Size(300, 250);
                    p2_character.Image = nbashow[1, 3];
                    p2_name.Text = nbaname[1, 3];
                }
            }

            private void characterb5_Paint(object sender, PaintEventArgs e)
//Yao_275x275
            {
                PictureBox pb = (PictureBox)sender;
                if (Convert.ToInt64(pb.Tag) == 1 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 3))
                {
                    e.Graphics.DrawRectangle(new Pen(Color.Red, 8), 0, 0, pb.Width,
pb.Height);
                    p1_character.Size = new Size(275, 275);
                    p1_character.Image = nbashow[1, 4];
                    p1_name.Text = nbaname[1, 4];
                }
                else if (Convert.ToInt64(pb.Tag) == 2 && flash == 1 ||
(Convert.ToInt64(pb.Tag) == 4))
                {
                    e.Graphics.DrawRectangle(new Pen(Color.Blue, 8), 0, 0, pb.Width,
pb.Height);
                    p2_character.Size = new Size(275, 275);
                    p2_character.Image = nbashow[1, 4];
                    p2_name.Text = nbaname[1, 4];
                }
            }
        }
}
*****************************************************************************************
```