南台科技大學 多媒體與電腦娛樂科學系

基礎程式設計 期中報告

三鍵反應遊戲

班級：多樂一甲

姓名：葉易先

學號：4B0K0110

目錄

1.簡介

本次基礎程式設計期中作業我將使用 Visual Studio(以下簡稱 VS)及 C#語言製作一款三按鍵輸入的反應遊戲，其玩法模仿自曾經的當紅手遊 ShotZombie。

ShotZombie 為日本的面白革命團隊於 2013 年推出的手機遊戲，簡單易懂的玩法與賦有競爭性的機制，使其迅速串紅。

圖-ShotZombie 商店頁面：簡單易懂且爽快的玩法，使其得高評價。

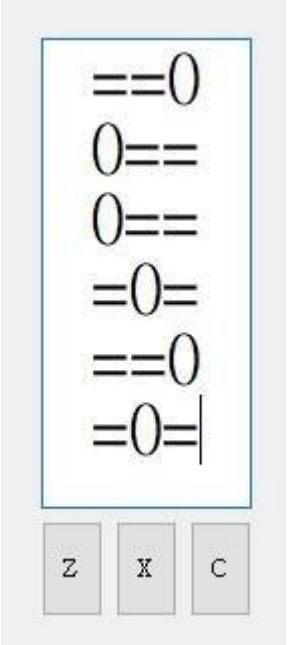
遊戲玩法：畫面上會出現位置隨機的目標，透過對應按鍵能消除最底端的目標，並讓上方的目標往下掉，打錯目標會暫時被禁止打擊目標，玩家須在最短時間內打擊所有目標。

2.構思&設計

2-1.畫面

在決定模仿對象後，考慮到技術與時間不足，無法用圖像還原其畫面呈現，因此選擇最簡單的方法：使用文字來還原模仿對象的畫面。

決定畫面表現方法後，便利用 VS 的 WinForm 設計功能做出簡單的畫面 Demo：



以 textbox 作為遊戲畫面，在其中生成位置隨機的目標「0」及作為背景的「=」，下方有三個對應文字的按鍵。

圖-遊戲介面 Demo

2-2.流程圖、遊戲機制

決定好畫面呈現方式後，開始構思程式碼的邏輯及流程，為了方便釐清邏輯並進行最佳化，需製作視覺化的流程圖。

我選擇使用名為 Draw.io 的線上流程圖製作工具，該工具允許用戶直接在瀏覽器上製作流程圖並下載可修改的存檔，是非常方便的線上工具。

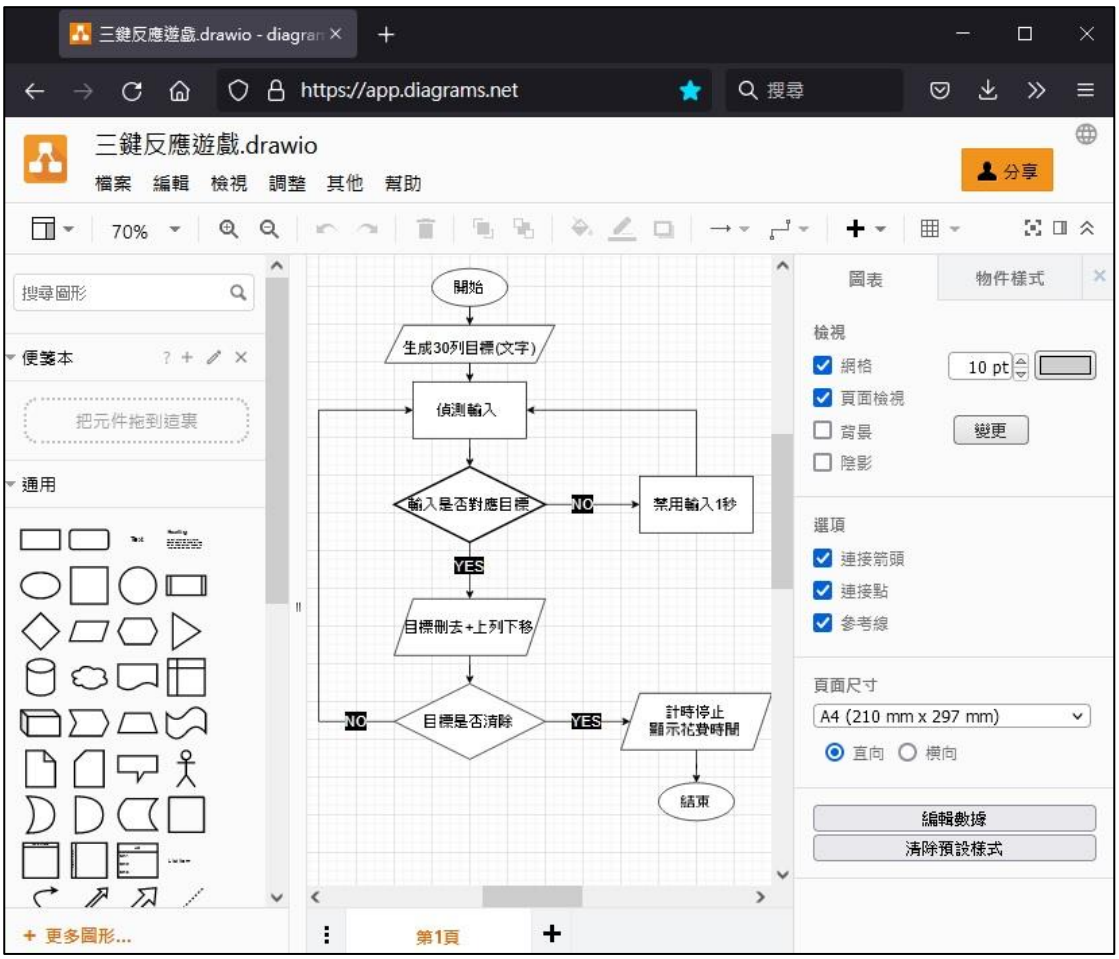


圖-Draw.io 網頁版：功能齊全的線上流程圖編輯器。

根據遊戲機制製作的簡易流程圖，其中包含從遊戲開始到結束的運作邏輯。

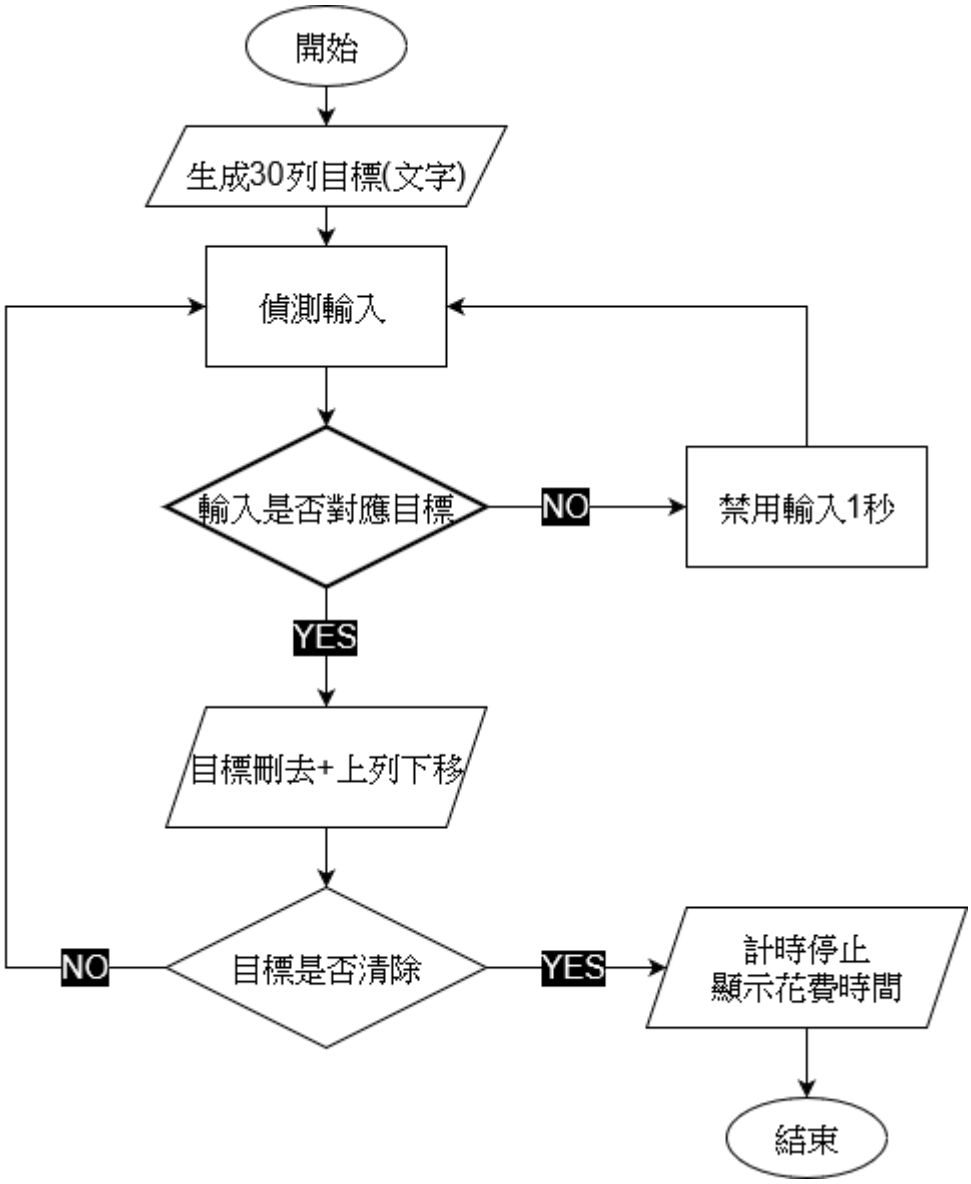


圖-簡易流程圖：藉由流程圖來明確程式碼的編寫方向。

為了簡化製作流程，修改了模仿對象的部分機制，將原本固定的遊戲時限改為無限制的計時器，而遊戲目的則從「時限內獲得最高分數」改為「最短時間內結束遊戲」。

3.實作

有了簡易的流程圖，便可開始依照流程圖編寫程式。

3-1.目標生成

依照流程圖，最先編寫的是打擊目標的生成。

```
//宣告目標(文字)
public string txt, txtA = "O==", txtB = "=O=", txtC = "==O"
//生成打擊目標
for (cnt = 1; cnt <= 30; cnt++)
{
   int tag = rnd.Next(1, 4);   //建立隨機變數

   switch (tag)                //根據隨機變數生成目標
     {
       case 1:
            textBox1.Text = textBox1.Text + "\r\n" + Convert.ToString(txtA);
            break;
       case 2:
            textBox1.Text = textBox1.Text + "\r\n" + Convert.ToString(txtB);
            break;
       case 3:
            textBox1.Text = textBox1.Text + "\r\n" + Convert.ToString(txtC);
            break;
     }
}
```

透過 switch 判斷式及隨機變數，便可在遊戲畫面中自動生成隨機的打擊目標。

3-2.目標判斷函數

依照流程圖，接下來應要處理輸入判斷，但因為輸入判斷中包含了目標判斷，因此優先製作目標判斷的函數。

```
public bool miss;              //懲罰狀態布林值
public void gameLogic(int pos) //目標判斷函數
{
                       //宣告文字框中的文字及文字的長度
  txtL = textBox1.Text.Length;
  txt = textBox1.Text;
                       //擷取文字框中對應按鍵的文字
  switch (txt.Substring(txtL - pos, 1))
   {
     case "O":         //擊中正確目標
                       //消除最底端目標
          textBox1.Text = txt.Substring(0, txtL - 5);
          txtL = textBox1.Text.Length;
                       //文字框內捲至最下方
          this.textBox1.Select(txtL, 0);
          this.textBox1.ScrollToCaret();
          break;
     case "=":          //擊中錯誤目標
          miss = true;  //懲罰狀態開啟
          DTcnt = 0;    //懲罰計時器中的計數器初值
                        //開啟懲罰計時器
          timeDelay.Enabled = true;
          break;
    }
}
```

該函數中使用到了懲罰計時器，因此接下來也優先製作懲罰計時器。

3-3.懲罰計時器

當玩家打錯目標時，遊戲將會禁止玩家輸入 0.5 秒作為懲罰，而該 0.5 秒將透過 timer 類別計算。

```
//懲罰計時器
private void timeDelay_Tick(object sender, EventArgs e)
{
  DTcnt++;                          //計數器逐Tick 增加

  double BT = 10 - DTcnt;
  DScnt = BT / 10;                  //顯示懲罰時間倒數
  label3.Text = "懲罰" + Convert.ToString(DScnt);

  if (DTcnt >= 5)                   //懲罰時間結束
    {
      timeDelay.Enabled = false; //關閉計時器
      miss = false;                 //關閉懲罰狀態
    }
}
```

該 timer 啟動後，會以每 100 毫秒(0.1 秒)為單位運作一次，並在畫面上顯示懲罰時間倒數，運作五次滿足 0.5 秒的條件後，便停止運作。

完成懲罰計時器後，整個目標判斷函數才算完成。

3-4.輸入判斷

完成了目標判斷函數後，便可開始製作輸入判斷。

```
//輸入判斷函數
private void Form1_KeyDown(object sender, KeyEventArgs e)
{
  if (e.KeyCode == Keys.Enter)
  {
    button4.PerformClick();    //Enter 觸發開始按鈕
  }

  if (miss == false)           //判斷懲罰狀態是否開啟
  {
    switch (e.KeyCode)
      {
        case Keys.NumPad1:     //按下數字鍵
            gameLogic(3);      //回傳該鍵對應位置值給函數
            break;             //以下同上
        case Keys.NumPad2:
            gameLogic(2);
            break;
        case Keys.NumPad3:
            gameLogic(1);
            break;
      }
  }
  else                         //如果懲罰狀態開啟
    return;                    //跳過該程序
}
```

輸入判斷寫在 Form 的 KeyDown 事件中，除了判斷玩家在遊戲中使用哪個鍵打擊目標，當玩家打錯目標時會將該程序跳過使玩家無法打擊，另外也添加了按下 Enter 即可開始遊戲的功能，使玩家能快速進行遊戲。

3-5.遊戲計時

該遊戲的重點在於考驗玩家反應速度，而最能反映速度快慢的是花費的時間，因此計時器的準確度非常重要。

然而 C#內建的 timer 類別會根據其執行內容複雜程度造成一定延遲，無法精確計算玩家花費的時間，因此該計時器將會借助電腦上的「系統時間」做計算。

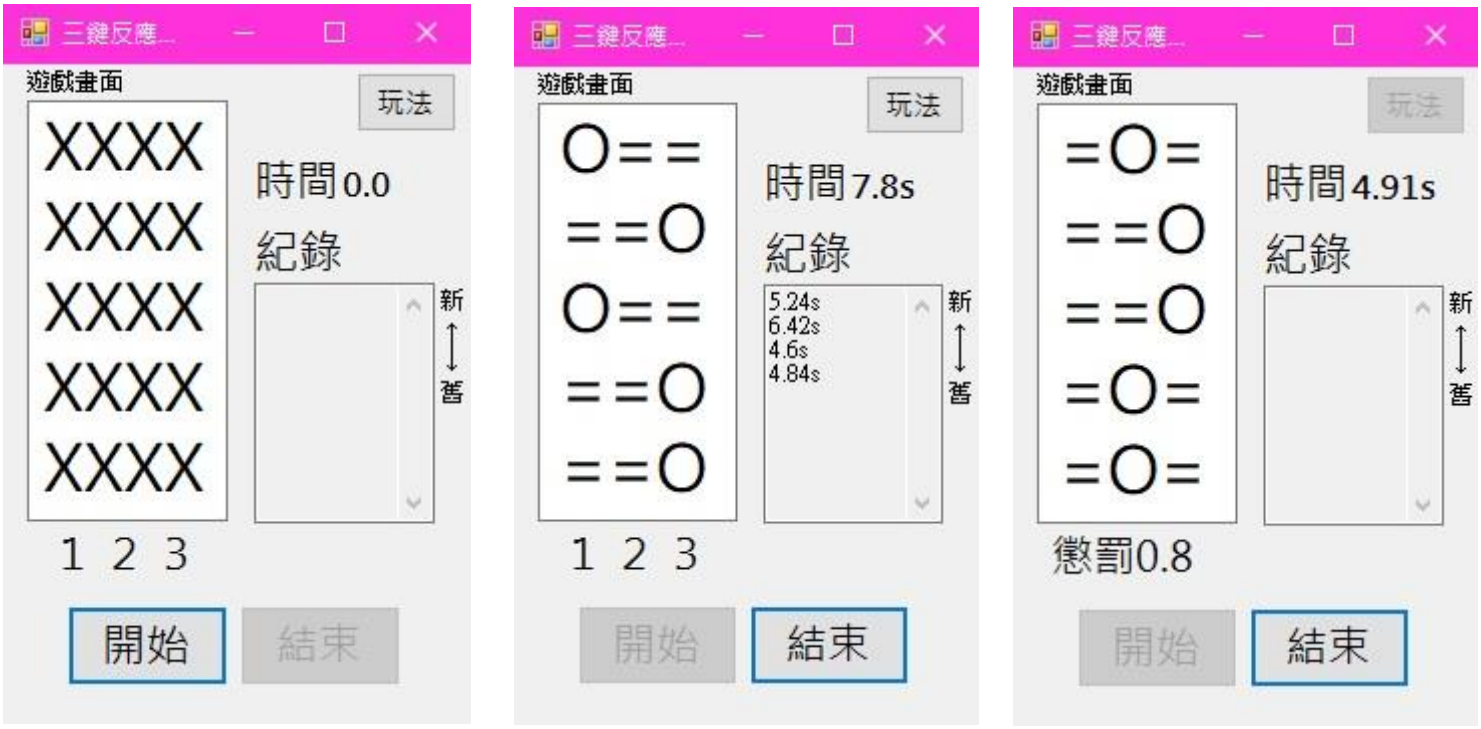
```
//遊戲計時
TimeSpan ST, nT, stT;
private void timerGame_Tick(object sender, EventArgs e)
{
  nT = DateTime.Now.TimeOfDay;          //擷取當下的系統時間
                                        //(遊戲開始時的時間已寫在「開始」按鈕中)
  ST = nT - stT;                        //計算「當下時間」與「開始時間」的時間差

  double S = Math.Round
            ((double)Convert.ToDouble(ST.TotalMilliseconds) / 1000, 2);

  label4.Text = Convert.ToString(S) + "s";
                                        //將時間差顯示在畫面上
  if (textBox1.Text.Length == 20)       //判斷目標是否全數清除
  {
    timerGame.Enabled = false;           //關閉遊戲計時
    textBox2.Text = Convert.ToString(S)+ "\r\n"+textBox2.Text;
  }                                     //將遊戲時間紀錄在榜上
}
```

雖然使用了 timer 類別，實際上計算時間的並不是 timer，而是電腦上的系統時間，藉由計算系統時間的時間差能得出最精準的遊戲時間，而 timer 的作用更像是迴圈。

5.最終成果&參考文獻



最終遊戲執行畫面

參考文獻：微軟開發者網路(MSDN)、Microsoft Docs、中國開發者網路(CSDN)、Stack Overflow、DelftStack、藍色小舖、巴哈姆特程式設計版、各路大神的部落格。