超級拉霸機

呂俐君

2020.08.07

一、動機

在構想階段我想盡可能的使用到不同種類的程式概念，然身為初學者，對於高階運算的程式一竅不通，因此決定採用迴圈、陣列、判斷的概念為主，搭配讓使用者因為無法完全控制而想一玩再玩的隨機函數，產生了製作拉霸機的想法。

「賭博」之所以令人著迷，是因為有可能以少量付出換取大筆收穫。因此我設計了兩種模式，一般模式及成功者模式：人們都希望自己永遠是幸運的，那麼在這個遊戲中可以選擇是否花多一些成本來保障自己的運氣。然而事情沒有絕對，在成功者模式中若沒有成功，相對的會扣掉更多的金幣。

我最主要是希望運用現有的程式知識，再自行延伸學習其他概念，以簡單的介面呈現出擁有各種結果的遊戲，做出具有吸引力的小遊戲。

第二次計分作業則是應用到函式和其他更多技巧，因此主要是在擴增功能：雙人遊戲模式，並且改善原程式碼使之更為人性化。

二、構想解說

流程介紹：

1. 進入遊戲
2. 選擇模式
    (1) 一般模式
    (2) 成功者模式
    (3) 雙人模式
        a. 模式
        b. 模式
        c. 模式
3. 使用該模式金幣數量
4. 執行Slot
5. 判斷遊戲結果
6. 執行對應增減金幣數量
7. 判斷是否繼續遊戲
8. 重新開始進入遊戲或結束

三、程式測試規劃

測試項目	通過標準	測試流程	測試結果	通過
隨機選擇及符號配對是否正確	所選數字需符合陣列位置	將隨機所選的數字及元素印出來	相符，然重複性高	O
將隨機函數增加範圍再除以8	使得數字選擇分布較為平均	以迴圈印出大量數字比對分佈機率	較平均	O
確認判斷式及敘述及相應結果是否相符	金幣數量合理、敘述正確	選擇各選項並檢查	有些遺漏的選項並無解決之道、金幣數不足卻仍可繼續遊戲	已修正
拉霸機呈現介面及操作是否正確	不必重複按enter、無不該出現之換行	印出結果檢查	有不當換行	已修正
使用console.write+console.clear迴圈	創造出亂碼的視覺效果	使用迴圈重複執行	Console.clear無法刪除字元，只會以空白清空頁面	待修正
按鍵操作控制	按任意鍵繼續遊戲，esc鍵退出	執行程式碼並嘗試各種特殊按鍵	成功	O
雙人模式	成功運作	尚未測試	目前是將大致流程寫出，細節仍需修正才可執行	待修正

四、結果與討論

1. 雙人模式初始設定與規則說明

```
27    //雙人模式
28    //初始設定與規則說明
29  Console.WriteLine("{0}，已經進入雙人模式了！趕快輸入另一位玩家的名字吧～",Name);
30  Console.WriteLine("二號玩家姓名：");
31  string Name2 = Console.ReadLine();
32  int life2 =10;
33  Console.WriteLine("好的，讓我開始說明規則吧！你們現在分別有10枚金幣，每局遊戲總共有三輪，每輪中有兩次遊戲，每次遊戲由兩位玩家輪流決定下注方式。");
34  Console.WriteLine("下注的方式有三種：{0}a. 決定自己該輪的金幣數量，則對方須付出相同數量之金幣，或不足者則需all in。");
35  Console.WriteLine("b.指定對方須付出的金幣數量，自己則至少須付出指定數量之2/3以上，條件：自己所擁有的金幣數量需為指定數目之三倍。");
36  Console.WriteLine("c. All in，則對方必須付出(1)自己擁有的1/2之金幣，或(2)至少相同數目之金幣(兩項規則中擇金幣數較高者執行)，不足者all in。");
37  Console.WriteLine("遊戲勝負責由三輪結束後金幣較多者獲勝！");
38  Console.WriteLine("OK！現在就馬上開始遊戲吧！")
```

(1) 建立第二位玩家身份

(2) 給予初始金幣（life2），並且顯示說明開始遊戲

(3) 進入While迴圈，使遊戲可以重複進行

2.遊戲開始

```
39      //遊戲輪次的迴圈
40      while (true)
41      {
42          for(int i=1;i<7;i++){
43          string WhosTurn = Name,SecondOne=Name2;
44          int WhosMoney=lifes,SecondMoney=life2,using1=0,using2=0;
45          int[] compare1 = new int[3];
46          int[] compare2 = new int[3];
47          //為啥用三元運算子不成功？(i%2==0)?(WhosTurn=Name2):(WhosTurn=Name);
48          //判斷是誰的輪次
49          if(i%2==0){WhosTurn=Name2;SecondOne=Name;WhosMoney=life2;SecondMoney=lifes;}
50          else WhosTurn=Name;SecondOne=Name2;WhosMoney=lifes;SecondMoney=life2;
51          Console.WriteLine("現在，是{0}決定下注方式的輪次:a. 決定自己該輪的金幣數量，則對方須付出相同
52          char way=char.Parse(Console.ReadLine());
```

(1) 以for迴圈執行一局六次的遊戲

(2) 設立兩個陣列（compare1&2）來儲存兩位玩家的遊戲結果

(3) 判斷本局主導者，並使其決定下注模式

2.1 選擇 a. 模式

```
53      //判斷選擇的下注模式
54      /*為啥用switch不成功？？
55      switch (way){
56          case 'a':
57      }*/
58      if(way=='a')||(way ='A'){
59          Console.WriteLine("{0}，你目前有{1}枚金幣，請問你要下注多少金幣:",WhosTurn,WhosMoney);
60          using1=Int32.Parse(Console.ReadLine());
61          if(SecondMoney<=using1){
62              using2=SecondMoney;
63              Console.WriteLine("{0},你只剩下{1}枚金幣，必須all in了，賭一把吧！",SecondOne,SecondMoney);
64          }
65          else{
66              Console.WriteLine("{0}，你目前有{1}枚金幣，並且必須使用掉{2}枚:",SecondOne,SecondMoney,using1);
67          }
68      }
```

(1) 若指定金額高於被指定者所擁有之金幣，則需以all in替代；

(2) 反之兩者付出相同數量之金幣

2.2 選擇 b. 模式

```
69      else if(way=='b')||(way='B'){
70          //選擇b的條件限制
71          if(lifes<3){
72              Console.WriteLine("金幣不足，無法執行b.方法，請重新選擇模式");
73              i--;
74              continue;
75          }
76          Console.WriteLine("{0}，你目前有{1}枚金幣，請問你要指定{2}下注多少金幣(最高可指定額度:{3}):",WhosTurn,WhosMoney,SecondOne,WhosMoney/3);
77          using2=Int32.Parse(Console.ReadLine());
78          while(using2>WhosMoney/3){
79              Console.WriteLine("(最高可指定額度是{0}枚耶！你要求的太多了！請重新指定{1}的下注金幣",WhosMoney/3,SecondOne);
80              using2=Int32.Parse(Console.ReadLine());
81          }
82          Console.WriteLine("{0}，你被指定的下注金額是{1}，因此你還剩餘{2}枚金幣",SecondOne,using2,SecondMoney-using2);
83          Console.WriteLine("{0}，你的下注金額則是{1}，因此你還剩餘{2}枚金幣",WhosTurn,using1,WhosMoney-using1);
84      }
```

(1) 首先判斷是否有資格進入b.模式，若無則需重新選擇

(2) 指定者輸入指定金額，並且以while迴圈確保其輸入為合理金幣數量，直到符合才得以break進入下一步

2.3 選擇 c. 模式

```
85      else if(way=='c')||(way=='C'){
86          using1=WhosMoney;
87          if(SecondMoney<=using1){
88              Console.WriteLine("{0},你只剩下{1}枚金幣，也必須all in了，賭一把吧！",SecondOne,SecondMoney);
89              using2=SecondMoney;
90          }
91          else if (SecondMoney/2<=using1){
92              using2=using1;
93              Console.WriteLine("{0},你必須下注{1}枚金幣。",SecondOne,using2);
94          }
95          else
96          {
97              using2=SecondMoney/2;
98              Console.WriteLine("{0},你必須下注{1}枚金幣。",SecondOne,using2);
99          }
100     }
101     //防止錯誤漏洞
102     else{
103         Console.WriteLine("輸入格式錯誤，請重新選擇模式");
104             i--;
105             continue;
106     }
```

(1) 以 if-else if 判斷被指定者之金幣數量，以三種條件中對指定者最有利之方式執行

(2) 若所選擇的下注模式格式錯誤，則需重新選擇，並不計算這次遊戲之計數 i

3. 開始執行比賽

```
107     //開始執行比賽
108     PlayTheSlot(compare1);
109     PlayTheSlot(compare2);
110
111     if(TypeOfTheResult(compare1)<TypeOfTheResult(compare2)){
112         WhosMoney-=using1;
113         SecondMoney+=using1;
114         Console.WriteLine("{0}贏得這輪遊戲！，你獲得了{1}枚金幣！",SecondOne,using1);
115         Console.WriteLine("{0}輸了，你失去{1}枚金幣。",WhosTurn,using1);
116     }
117     else if(TypeOfTheResult(compare1)>TypeOfTheResult(compare2)){
118         SecondMoney-=using2;
119         WhosMoney+=using2;
120         Console.WriteLine("{0}贏得這輪遊戲！，你獲得了{1}枚金幣！",WhosTurn,using2);
121         Console.WriteLine("{0}輸了，你失去{1}枚金幣。",SecondOne,using2);
122     }
123     else{
124         Console.WriteLine("兩位平手，維持現狀")
125     }
126     if(lifes<=0)||(life2<=0){
127         string winner;
128         if (lifes<=0) winner=Name2;
129         else winner=Name;
130         Console.WriteLine("目前{0}有{1}枚金幣，而{2}則有{3}枚金幣，遊戲提前結束。",Name,lifes,Name2,life2);
131         Console.WriteLine("贏家是...恭喜{0}！！！",winner);
132         break;
133     }
134     Console.WriteLine("目前{0}有{1}枚金幣，而{2}則有{3}枚金幣，這局還剩下{4}次遊戲。",Name,lifes,Name2,life2,6-i);
```

(1) 將前一次的「一般模式」寫成函式（PlayTheSlot），並且分別將結果存入兩位參賽者之陣列中

(2) 將前一次的「遊戲結果判斷」寫成函式（TypeOfTheResult），並且輸出為數值

(3) 比較參賽者們遊戲結果之數值，增減其金幣數量

4. 整局遊戲結果判斷

```
137             //本局的遊戲贏家
138             if(lifes<=0)||(life2<=0) break;
139             else if(lifes>life2){
140                 Console.WriteLine("{0}有{1}枚金幣，而{2}則有{3}枚金幣，因此本局的贏家是...。",Name,lifes,Name2,life2,Name);
141             }
142             else if(lifes<life2){
143                 Console.WriteLine("{0}有{1}枚金幣，而{2}則有{3}枚金幣，因此本局的贏家是...。",Name,lifes,Name2,life2,Name2);
144             }
145             else{
146                 Console.WriteLine("恭喜兩位平手～");
147             }
```

(1) 比較兩位參賽者之總金幣數並判斷勝負，給予回饋

4. 判斷是否繼續遊戲

```
148             //是否重複遊戲
149             Console.WriteLine("按下任意鍵重新開始，或按下esc結束遊戲");
150             ConsoleKeyInfo cki;
151             Console.TreatControlCAsInput = true;
152             cki = Console.ReadKey();
153             if (cki.Key == ConsoleKey.Escape) {
154                 Console.WriteLine("遊戲結束");
155                 break;
156             }
157             else continue;
158             }
```

(1)以ConsoleKey讀取使用者按鍵內容，並使其選擇重新進入迴圈開始遊戲，或是break結束遊戲。

五、學習心得

根據上次列舉出的未來展望的執行成果如下：

1. 使得操作更直觀，以esc/enter/左右鍵選擇模式來開始、結束遊戲。（成功）
2. 使得每個符號在跑出來時，能夠有亂數轉動的視覺效果，並進一步增加遊戲介面，有彩色拉霸機與投金幣等等的行為。（嘗試許久，尚未成功）
3. 讓使用者可以根據自己的喜好按下停止的時刻，增加互動性。（與2.連動）

本次面臨的問題與感受：

1. 嘗試寫上述第二點時，因為Console.Clear的執行模式與預期中的不同，並且找不到可以刪除「已列印出之字元」的程式碼，而卡住許久，但卻也意外找到許多可以刪減原字串內容的方式，下次會想要以「移動游標」的方式繼續嘗試。
2. 這次的命名因為有更明確而在撰寫過程中讓邏輯運轉快了許多，沒有搞混的狀況。
3. 多層的if-else判斷式會許多程式碼重複的問題；而多層迴圈無法同時break，反而需要在外層迴圈再判斷一次，也是非常的冗長，應該會有更好的辦法。

未來展望：

1. 將雙人模式完成，修復錯誤。
2. 將部分程式碼轉換為函式呼叫，使得主要程式碼邏輯架構更簡潔明瞭。
3. 使得每個符號在跑出來時，能夠有亂數轉動的視覺效果，並進一步增加遊戲介面，有彩色拉霸機與投金幣等等的行為。
4. 讓使用者可以根據自己的喜好按下停止的時刻，增加互動性。

這一次改成以esc及任意鍵操作後確實直觀很多，很有成就感；但視覺效果尚未成功卻花了大量時間就有點可惜，因為無法呈現在報告中，看出時間佔比。

除此之外，光是建構好比較複雜的邏輯架構就花了蠻長的時間，但因為很希望可以增加遊戲的樂趣而想辦法增加遊戲的選項及變化，下一次主要先將雙人模式中的細節修正，可執行後再進行其他部分的優化。

六、參考文獻

1. NTU COOL 課程影片
2. Microsoft- Documentation

   Readkey, Trim, Console.clear, function, void