CHAPTER 18

視窗遊戲應用程式專題實作

- 拉霸遊戲機實作
- 記憶大考驗實作

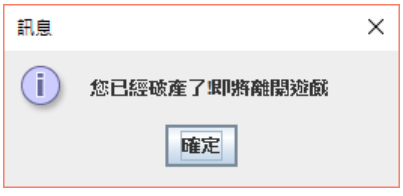
18.1 拉霸遊戲機實作

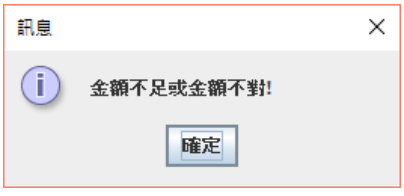
拉霸遊戲機是坊間電玩機常見的機台，常見有九個圖示的拉霸機與三個圖示的拉霸機，九個圖示的拉霸機得獎的機率與設計的過程比較複雜，本節介紹三個圖示的拉霸機。如下圖：

一. 系統功能說明

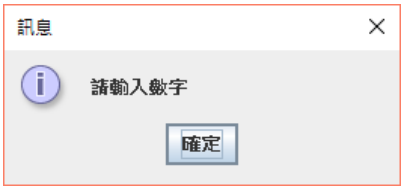
下列是拉霸遊戲機的遊戲規則說明：

1. 開始時必須先設定本次投注的數量，接著按 下注 鈕即開始玩拉霸機。若投注量為 0，出現右下圖對話方塊顯示 "您已經破產了！即將離開遊戲" 訊息並結束遊戲；若投注量超過擁有的總數量或投注量小於等於 0，即出現左下圖對話方塊顯示 "金額不足或金額不對! " 訊息。投注總量預設 50。

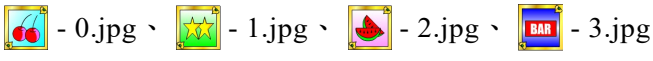




2. 若拉霸遊戲的投注量文字方塊輸入文字資料並按 下注 鈕，則出現下圖對話方塊顯示 "請輸入數字" 訊息。



3. 若允許投注並按 下注 鈕，此時拉霸機開始啟動且視窗上的三個圖會以亂數的方式由下面 0.jpg~3.jpg 四張圖檔中任選一張來顯示。



大約過三秒後，拉霸機停止轉動，再判斷是否有中獎？中獎條件如下：

① 若得到 3 個  圖，則依投注量得到 3 倍。

② 若得到 3 個  圖，則依投注量得到 10 倍。

③ 若得到 3 個  圖，則依投注量得到 20 倍。

④ 若得到 3 個  圖，則依投注量得到 50 倍。

二. 輸出入介面設計

1. 定義 MyJFrame 類別繼承自 JFrame 類別(視窗)，由於此類別中的 下注 鈕必須處理 ActionEvent(按一下事件)，因此 MyJFrame 還必須實作 ActionListener 介面的 actionPerformed 方法。

2. 在 MyJFrame 類別的建構式內加入下列元件：

   ① 建立 icon[0]~icon[3] 四個 ImageIcon 用來存放櫻桃、星星、西瓜、bar 四個拉霸機圖示，這四個圖示是在專案 barImg 資料夾下的 0.jpg、1.jpg、2.jpg、3.jpg。

   ② 建立 jlbl[0]~jlbl[2] 三個標籤用來顯示拉霸遊戲的三個圖，即 jlbl[0]~jlbl[2] 三個標籤用來顯示 icon[0]~icon[3] 所代表的圖。

   ③ 建立 jlblSum 標籤用來顯示 "總數量：" 訊息。

   ④ 建立 jlblBetting 標籤用來顯示 "投注量：" 文字訊息。

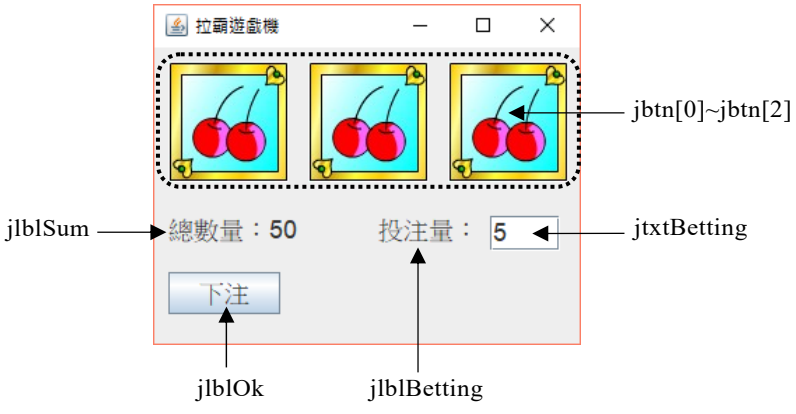
   ⑤ 建立 jtxtBetting 文字方塊用來設定每一次拉霸要投注的數量。

   ⑥ 建立 jbtnOk 下注 鈕用來啟動拉霸機。

   ⑦ 將 MyJFrame 視窗大小設為寬 320、高 250。

   ⑧ 將 0.jpg~3.jpg 圖檔放到目前專案的 barImg 資料夾下。

   視窗內建立的元件如下：



三. 問題分析

1. 拉霸機上面三個圖示如何建立？

   本例使用標籤元件來顯示拉霸遊戲上的圖示，因此在視窗上建立陣列元件 jlbl，其陣列元素為 jlbl[0]~jlbl[2] 用來顯示三個拉霸圖。為方便使用迴圈將圖片指定給 jlbl[0]~jlbl[2] ，必須建立陣列元件 icons，其陣列元素為 icons[0]~icons[3] 用來存放  - 0.jpg、  - 1.jpg、  - 2.jpg、  - 3.jpg 四張圖檔，此時即可使用迴圈配合 jlbl[0]~jlbl[2] 以亂數方式隨機顯示 icons[0]~icons[3] 的圖示。

2. 如何應用執行緒由 jlbl[0]~jlbl[2] 陣列元素(標籤元件)中亂數取圖？

   當按  鈕使拉霸機啟動，此時執行緒物件啟動執行 run()方法，在 run() 方法中設定 jlbl[0]~jlbl[2] 陣列元素分別以亂數方式由 icons[0]~icons[3] (即 0.jpg~3.jpg) 四張圖片中選取一張來顯示。為了讓拉霸機上面的三張水果圖有滾動的感覺，該執行緒物件每隔 0.1 秒進入休眠狀態並重新亂數取圖一次，直到連續 10 次才停止。

3. 如何判斷是否中獎？

   當拉霸機的 jlbl[0]~jlbl[2] 陣列元素停止換圖時，即馬上判斷是否中獎，此處將四張圖分別設定代碼以便判斷所中的獎項及倍數，如下：

   ①  荔枝代碼為 0

   ②  星星代碼為 1

   ③  西瓜代碼為 2

   ④  BAR 代碼為 3

   n[0]=1　　n[1]=2　　n[2]=0

   譬如：上圖為拉霸機亂數出來的水果圖，將三個水果圖代碼依序存入陣列 n，當陣列元素 n[0]、n[1] 和 n[2] 的代碼都相同表示有中獎，依中獎規則的指定倍數賠。

四. 事件流程

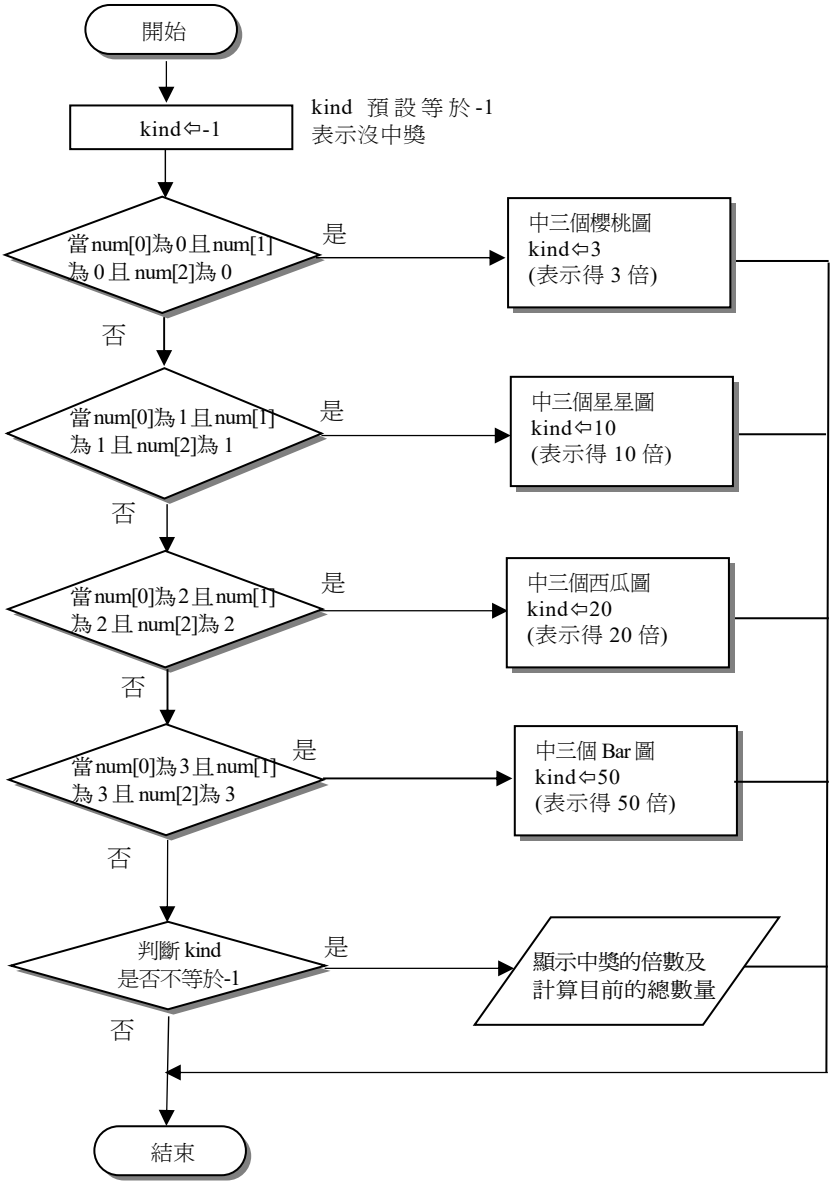
在 下注 鈕 ActionListener 傾聽者物件的 actionPerformed 方法(即按 下注 鈕所執行的方法) 內啟動執行緒物件，該執行緒物件的 run()方法做下列指定事情：

1. 判斷投注量是否有誤？
   ① 若 sum 總數量等於 0 即表示沒有可用的投注額，此時出現對話方塊並顯示 "您已經破產了!即將離開遊戲" 訊息，接著馬上離開遊戲；若 sum 總數量不為 0 則繼續下一步驟。
   ② 由 jtxtBetting 文字方塊取得使用者的投注額並指定給 betting，接著判斷 sum 總數量是否小於 betting 投注額或 betting 投注額是否小於 0，若其中之一成立表示金額不足，此時出現對話方塊並顯示 "金額不足或金額不對!" 訊息，接著馬上執行 return 敘述離開事件方法；由於使用者可能在 jtxtBetting 文字方塊內輸入文字資料，因此此處請使用 try…catch…括住，以便補捉例外。
   ③ 當使用者在 jtxtBetting 文字方塊內輸入文字資料即產生執行時期的例外，此時出現對話方塊顯示 "請輸入數字" 訊息，提示使用者投注額輸入錯誤。
2. 如何每 0.1 秒讓 jlbl[0]~jlbl[2] 隨機顯示 icons[0]~icons[3] 的圖(即 0.jpg ~3.jpg)：
   ① 預設 k=0，進入 do…while 迴圈內。
   ② 使用 for 迴圈配合亂數使 jlbl[0]~jlbl[2] 隨機顯示 icons[0] ~icons[3] 的圖(即 0.jpg~3.jpg)，並將亂數產生的三個水果圖代碼依序存入陣列元素 n[0]、n[1] 和 n[2] 中。
   ③ 使 k 加 1，用來表示 do…while 迴圈執行的次數。
   ④ 使用 Thread.currentThread().sleep(100)敘述讓目前執行緒暫停 0.1 秒，使 jlbl[0]~jlbl[2] 能顯示指定的圖檔 0.1 秒。
   ⑤ 判斷 k 是否小於 10，若成立即跳到本項目步驟②繼續執行，否則離開 do…while 迴圈。

由於呼叫執行緒的 sleep()方法會產生 InterruptedException 的例外，因此請將上述程式碼寫在 try…catch…敘述內。程式碼如下：

```
        int k = 0;
        ......
        try {
            do {
                // 產生 0~3 之間的亂數並指定給 n[0]~n[2]
                // 並在 jlbl[0]~jlbl[2]隨機顯示櫻桃,星星,西瓜,bar 圖示
                for (int i = 0; i < jlbl.length; i++){
                    n[i] = (int)Math.round(Math.random() * 3);
                    jlbl[i].setIcon(icons[n[i]]);
                }
                k++;
                Thread.currentThread().sleep(100);//目前執行緒暫停 0.1 秒
        } while(k < 10);        // 若 k 大於 0，則停止拉霸遊戲
    } catch(InterruptedException ex) { }
```

3. 如何判斷中到那個獎項，流程圖如下：



五. 完整程式碼

由於本例會讀取 barImg 資料夾下的 0.jpg~3.jpg 圖檔，因此請將 barImg 資料夾置入目前專案下，本例完整程式碼及註解說明如下：

程式碼

檔名：\ex18\src\bar\Program.java

```
01 package bar; //置於 bar 套件下
02
03 import java.awt.*;          //使用 Font 類別請匯入 java.awt.*套件
04 import java.awt.event.*;    //使用事件請匯入 java.awt.event.*套件
05 import javax.swing.*;       //使用 swing 元件請匯入 javax.swing.*套件
06
07 // MyJFrame(拉霸遊戲視窗)繼承 JFrame 視窗元件
08 // MyJFrame 實作 ActionListener 介面的 actionPerformed 方法用來處理按鈕的按一下事件
09 class MyJFrame extends JFrame implements ActionListener {
10   // 宣告 jlbl[0]~jlbl[2]用來當拉霸遊戲三個圖
11   private JLabel[] jlbl = new JLabel[3];
12   // 宣告 icon[0]~icon[3] 用來存放櫻桃, 星星, 西瓜, bar 四個圖示
13   // 四個圖示依序為 0.jpg, 1.jpg, 2.jpg, 3.jpg
14   private ImageIcon[] icons = new ImageIcon[4];
15   // 宣告 jlblSum 標籤用來顯示 "總數量：" 訊息
16   // 宣告 jlblBetting 標籤用來顯示 "投注量：" 訊息
17   private JLabel jlblSum, jlblBetting;
18   // 宣告 jtxtBetting 文字方塊用來讓使用者輸入投注量
19   private JTextField jtxtBetting;
20   // 宣告 jbtnOk "下注" 按鈕
21   private JButton jbtnOk;
22   private int sum = 50;
23   //建構式
24   MyJFrame()
25   {
26     // 不使用版面配置
27     super.setLayout(null);
28     // 視窗標題設為 "拉霸遊戲機"
29     super.setTitle("拉霸遊戲機");
30     // 設定 icons[0]~icons[3]元件的圖示為 barImg 資料夾下的 0.jpg~3.jpg
31     for(int i = 0; i < icons.length; i++) {
32       icons[i] = new ImageIcon
                    (".\\barImg\\" + String.valueOf(i) + ".jpg");
33     }
```

```
34      // 建立 jlbl[0]~jlbl[2]，並指定三個標籤為櫻桃圖(0.jpg)，最後放入視窗內
35      for (int i = 0; i < jlbl.length; i++) {
36        jlbl[i] = new JLabel();
37        jlbl[i].setBounds(i*100+10, 10, 86, 86);
38        jlbl[i].setIcon(icons[0]);
39        add(jlbl[i]);
40      }
41      // 在視窗放入 jlblSum 標籤，該標籤顯示 "總數量："
42      jlblSum = new JLabel("總數量：" + String.valueOf(sum));
43      // 設定 jlblSum 標籤 x 座標 10, y 座標 120, 寬 160, 高 20
44      jlblSum.setBounds(10, 120, 160, 20);
45      jlblSum.setFont(new Font("微軟中黑體",Font.PLAIN, 18));
46      add(jlblSum);
47      // 在視窗放入 jlblBetting 標籤，該標籤顯示 "投注量："
48      jlblBetting = new JLabel("投注量：");
49      jlblBetting.setBounds(160, 120, 80, 20);
50      jlblBetting.setFont(new Font("微軟中黑體",Font.PLAIN, 18));
51      add(jlblBetting);
52      // 在視窗放入 jtxtBetting 文字方塊，讓使用輸入投注量
53      jtxtBetting = new JTextField();
54      jtxtBetting.setBounds(240, 120, 50, 25);
55      jtxtBetting.setFont(new Font("微軟中黑體",Font.PLAIN, 18));
56      add(jtxtBetting);
57      // 在視窗放入 jbtnOk 下注鈕
58      jbtnOk = new JButton("下注");
59      jbtnOk.setBounds(10, 160, 80, 30);
60      jbtnOk.setFont(new Font("微軟中黑體",Font.PLAIN, 18));
61      add(jbtnOk);
62
63      // 指定 jbtnOk 下注鈕的傾聽者為目前的物件
64      // 因此按下注鈕時會執行目前類別的 actionPerformed 方法
65      jbtnOk.addActionListener(this);
66
67      // 設定視窗大小為寬 320, 高 250
68      setSize(320, 250);
69      // 顯示視窗
70      setVisible(true);
```

```
71      // 設定按視窗的關閉鈕會結束程式
72      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
73    }
74
75    // 實作 ActionListener 介面的 actionPerformed 方法
76    public void actionPerformed(ActionEvent evt) {
77      // 建立執行緒 t 物件，並傳入 Runnable 介面物件
78      // 此執行緒用來啟動拉霸遊戲
79      // 讓 jlbl[0]~jlbl[2]以亂數方式顯示櫻桃, 星星, 西瓜, bar 四個圖示
80      // 並判斷是否中獎
81      Thread t = new Thread (
82        new Runnable() {
83          //實作 Runnable 介面的 run 方法
84          public void run() {
85            // k 用來計算拉霸遊戲的換圖次數
86            // kind 用來表示中獎倍數，kind 等於-1 表示沒中獎
87            int k = 0, kind = -1;
88            //n[0]~n[2] 用來存放產生的亂數值
89            int[] n = new int[jlbl.length];
90            int betting = 0;          // 用來存放投注量
91            try {
92              // 若 sum 總數量等於 0，表示沒有可用的投注額即離開遊戲
93              if(sum == 0) {
94                JOptionPane.showMessageDialog
                          (null, "您已經破產了!即將離開遊戲");
95                System.exit(0);
96              }
97              // 取得使用者的投注額，並指定給 betting
98              betting = Integer.parseInt(jtxtBetting.getText());
99              // 當總數量小於投注額或投注額小於 0，表示金額不足
100             if (sum < betting || betting <= 0) {
101               JOptionPane.showMessageDialog
                            (null, "金額不足或金額不對!");
102               return ;
103             }
104             sum -= betting;
105             jlblSum.setText("總數量：" + String.valueOf(sum));
106             // 按下注鈕啟動拉霸遊戲機後馬上即停用下注鈕
```

```
107                 // 防止使用者重複按下
108                 jbtnOk.setEnabled(false);
109              } catch(Exception ex) {
110                 JOptionPane.showMessageDialog(null, "請輸入數字");
111                 return ;
112              }
113              try {
114                 do {
115                    // 產生 0~3 之間的亂數並指定給 n[0]~n[2]
116                    // 並在 jlbl[0]~jlbl[2]隨機顯示櫻桃,星星,西瓜,bar 圖示
117                    for (int i=0; i<jlbl.length; i++) {
118                       n[i] = (int)Math.round(Math.random()*3);
119                       jlbl[i].setIcon(icons[n[i]]);
120                    }
121                    k++;
122                    //目前執行緒暫停 0.1 秒
123                    Thread.currentThread().sleep(100);
124                 } while(k < 10); //若 k 大於 0，則停止拉霸遊戲
125              } catch(InterruptedException ex) { }
126              // 判斷中那個獎
127              if (n[0] == 0 && n[1] == 0 && n[2] == 0) {
128                 kind = 3;     //三個圖為櫻桃, 得 3 倍
129              } else if(n[0] == 1 && n[1] == 1 && n[2] == 1) {
130                 kind = 10;    //三個圖為星星, 得 10 倍
131              } else if(n[0] == 2 && n[1] == 2 && n[2] == 2) {
132                 kind = 20;    //三個圖為西瓜, 得 20 倍
133              } else if(n[0] == 3 && n[1] == 3 && n[2] == 3) {
134                 kind = 50;    //三個圖為 bar, 得 50 倍
135              }
136              // 判斷是否中獎，若 kind 不等於-1 表示中獎
137              if (kind != -1) {
138                 JOptionPane.showMessageDialog(null,
                           "中獎得" + String.valueOf(kind) + "倍");
139                 // 目前總數量(總投注額)累加中獎數量
140                 sum += kind*betting;
141                 jlblSum.setText("總數量：" + String.valueOf(sum));
142              }
```

```
143                 jbtnOk.setEnabled(true); // 下注鈕啟用
144             }
145         });
146         t.start();//啟動執行緒，使拉霸機啟動，此時 jlbl[0]~jlbl[2]即以亂數秀圖
147     }
148 }
149 // 主程式
150 public class Program {
151     public static void main(String[] args){
152         // 建立 MyJFrame 視窗(拉霸遊戲)
153         new MyJFrame();  //執行第 9~148 行建立 MyJFrame 類別物件(拉霸遊戲物件)
154     }
155 }
```

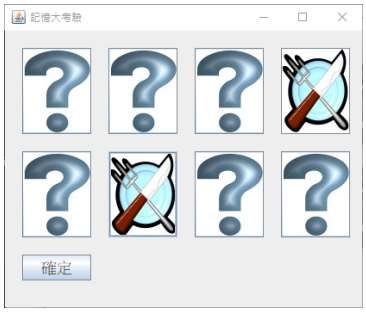
18.2 記憶大考驗實作

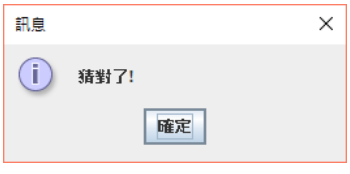
記憶大考驗遊戲在平板電腦及智慧型手機遊戲上是最常見的多媒體小遊戲。按 確定 鈕進行隨機配置圖片，且所有圖片會覆蓋並以問號圖顯示，玩家可使用滑鼠點選要翻開的圖片。當連續翻開兩張圖是相同時，表示完成配對一組圖片，當四組圖片皆被翻開後表示過關。

一. 系統功能說明

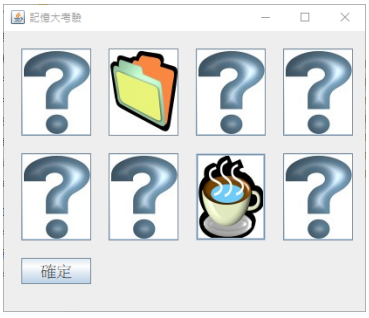
規則說明如下：

1. 連續翻開兩張相同的圖時，會出現對話方塊並顯示 "猜對了!" 訊息。當配對一組圖片之後，該圖片即失效不啟用。

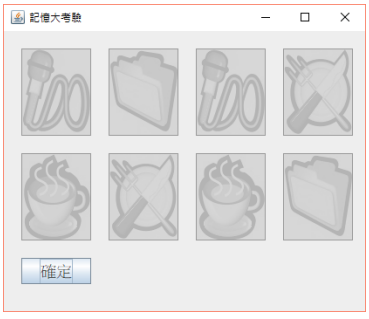




2. 連續翻開兩張不相同的圖時，會出現對話方塊並顯示 "不對哦!" 訊息。



3. 當四組圖片皆被翻開後，會出現對話方塊並顯示 "全對了…ya!" 訊息。



二. 輸出入介面設計

1. 定義 MyJFrame 類別繼承自 JFrame 類別(視窗)。

2. 在 MyJFrame 類別的建構式內加入下列元件：

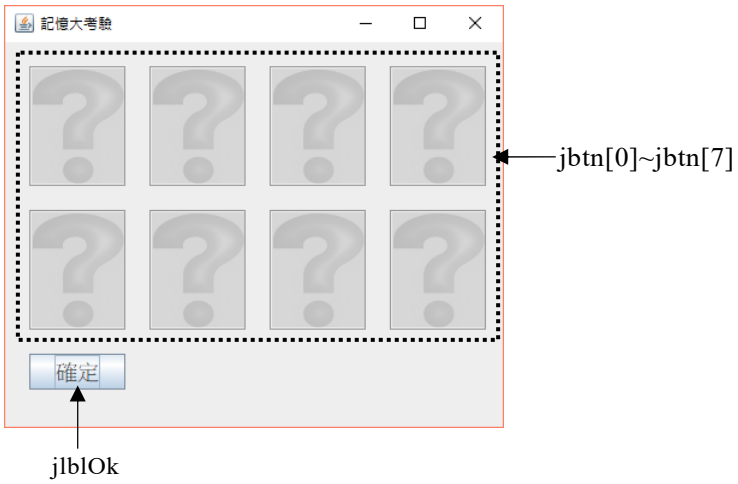
   ① 建立 icon[0]~icon[4] 四個 ImageIcon，用來存放 memoryImg 資料夾下的 0.jpg、1.jpg、2.jpg、3.jpg 及 4.jpg。

   ② 建立 jbtn[0]~jbtn[7] 八個按鈕陣列元件，用來當做記憶大考驗遊戲的 8 個圖示按鈕。

   ③ 建立 jbtnOk 

鈕用來啟動記憶大考驗遊戲。

   視窗內建立的元件如下：



三. 問題分析

1. 宣告與建立下列成員：

① 建立 icons[0]~icon[4] 五個 ImageIcon 陣列元件，用來存放專案的 memoryImg 資料夾下的 0.jpg(? 問號圖)、1.jpg(餐具)、2.jpg(咖啡)、3.jpg(資料夾)、4.jpg(麥克風)五個圖示。圖示如下：

② 建立 jbtn[0]~jbtn[7] 按鈕陣列元件，jbtn[0]~jtn[7] 是用來讓使用者玩記憶大考驗的八個圖示按鈕。

③ 建立 jbtnOk 確定鈕。宣告 jbtnf 表示第一次按下的圖示按鈕(即第一次翻開的圖片)、宣告 jbtns 表示第二次按下的圖示按鈕(即第二次翻開的圖片)。

④ 宣告 f 用來存放第一次按下圖片按鈕(翻開圖片)所取得的字串，宣告 s 用來存放第二次按下圖片按鈕所取得的字串。

⑤ 宣告 num 表示按下圖片按鈕的次數，若 num 等於 2 即判斷所翻開的兩張圖是否相同；win 表示共猜對幾組圖片，因為有 8 張圖片，所以必須翻開四組相同的圖片，因此 win 為 4 時表示過關。

⑥ 建立 rnd[0]~rnd[7] 陣列元素用來存放記憶大考驗每張圖所代表的編號。陣列元素的值相同表示為一對，若 rnd 各陣列元素值如下，則 rnd[0] 和 rnd[5] 其值為 0 即為一對，rnd[1]和 rnd[7] 其值為 1 即為一對，…其他以此類推。

```
rnd[0] = 0 ;  rnd[1] = 1 ; rnd[2] = 3 ; rnd[3] = 2 ;
rnd[4] = 2 ;  rnd[5] = 0 ; rnd[6] = 3 ; rnd[7] = 1 ;
```

2. 在視窗內建立 確定 鈕並指定該鈕 ActionEvent 的事件傾聽者物件，並在該傾聽者物件的 actionPerformed 方法做下列事情：

① 建立 ary[0]~ary[7] 用來存放圖示的編號，編號相同的為同一組。寫法如下：

```
int[] ary = new int[]{1,1,2,2,3,3,4,4};
```

② 宣告 n 變數預設為 0 用來存放產生的亂數。

③ 宣告 max 變數存放陣列索引上限 7。

④ 使用迴圈將 ary[0]~ary[7] 以亂數方式隨機置入 rnd[0]~rnd[7]內，接著將 rnd[0]~rnd[7]內的值放入 jbtn[0]~jbtn[7] 的 ActionCommand 內，以便將來進行比對翻開兩個圖片按鈕的值是否一樣，最後再將 jbtn[0]~jbtn[7] 按鈕的圖片設為 icons[0](即 0.jpg 問號圖)以及設為啟用。

上述步驟程式碼如下：

```
jbtnOk.addActionListener(new ActionListener(){  //按下確定鈕執行此處
    public void actionPerformed(ActionEvent evt) {
        // 建立 ary[0]~ary[7] 用來存放圖示的編號，編號相同的為同一組
        int[] ary = new int[]{1,1,2,2,3,3,4,4};
        int n = 0;   // 用來存放產生的亂數
        int max = ary.length-1;
        // 使用迴圈 jbtn[0]~jbtn[7]進行亂數存放 1.jpg~4.jpg
        // 編號相同為同一組
```

```
            for (int i = 0; i < ary.length; i++){
                n = (int)Math.round((Math.random() * max));
                rnd[i] = ary[n];
                ary[n] = ary[max];
                max--;
                jbtn[i].setActionCommand(String.valueOf(rnd[i]));
                jbtn[i].setToolTipText(String.valueOf(i));
                jbtn[i].setIcon(icons[0]);
                jbtn[i].setEnabled(true);
             }
        }
    });
```

3. 使用迴圈設定圖片按鈕排列成兩行，一行四個圖片按鈕；設定 jbtn[0]~jbtn[7] 顯示 q.jpg 問號圖示；設定 jbtn[0]~jbtn[7] 的圖片按鈕失效不啟用；指定 jbtn[0]~jbtn[7] 的 ActionEvent 的事件傾聽者物件。其程式寫法如下：

```
        int x = 0, y = 0;
        for (int i = 0; I < jbtn.length; i++){
        jbtn[i] = new JButton();
        jbtn[i].setBounds(x * 100 + 20, y * 120 + 20, 80, 100);
        jbtn[i].setIcon(icons[0]);        // 按鈕預設顯示 ? 問號圖
        jbtn[i].setEnabled(false);
        x++;
        if (i % 4 == 3){
            y++;
            x = 0;
        }
        //指定 jbtn[0]~jbtn[7]的事件傾聽者物件
       jbtn[i].addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent evt) {
            //事件處理相關程式碼
        }
    });
```

4. 指定 jbtn[0]~jbtn[7]的匿名事件傾聽者物件，並在傾聽者物件的 actionPerformed 方法做下列事情：

   ① 先將 num 加 1。

   ② 判斷 num 是否為 1？若成立表示第一次按下圖片按鈕，此時按下的圖片按鈕即顯示目前翻開的圖示，並使用 getActionCommand()方法取得按鈕所代表的字串並指定給 f，使用 getSource()方法取得目前按下的按鈕並指定給 jbtnf；若不成立則跳到步驟③。

   ③ 判斷 num 是否為 2？若成立表示第二次按下圖片按鈕，此時請做下面事情：

   - 按下的圖片按鈕顯示目前翻開的圖示，使用 getActionCommand()方法取得按鈕所代表的字串並指定給 s，使用 getSource()方法取得目前按下的按鈕並指定給 jbtns。
   - 判斷 f 是否等於 s 且 jbtnf 不等於 jbtns？若不成立則跳到下一步驟。若成立表示連續翻開兩張圖片一樣並做下列事情：
     - 出現對話方塊顯示 "猜對了!" 訊息。
     - 將第一次和第二次翻開的圖片按鈕設為失效不啟用。
     - win 加 1 表示翻開一組相同的圖片。
     - 判斷 win 是否等於 4？若成立表示 4 組相同的圖示皆翻開，此時出現對話方塊顯示 "全對了...ya!" 訊息。
   - 若 f 不等於 s 或 jbtnf 等於 jbtns？即表示連續翻開兩張的圖片不相同。此時出現對話方塊並顯示 "不對哦" 的訊息，最後在第一次 jbtnf 和第二次 jbtns 翻開的圖片按鈕顯示 icons[0] (0.jpg 問號圖示)表示將牌蓋住。
   - 將 s 和 f 字串設為空白，num 設為 0，表示要重新設定一組新翻開的圖片。

四. 完整程式碼

由於本例會讀取 memoryImg 資料夾下的 0.jpg~4.jpg 圖檔，因此請將 memoryImg 資料夾置入目前專案下，本例完整程式碼及註解說明如下：

程式碼

檔名：\ex18\src\memory\Program.java

```
01 package memory;
02
03 import java.awt.*;          //使用 Font 類別請匯入 java.awt.*套件
04 import java.awt.event.*;  //使用事件請匯入 java.awt.event.*套件
05 import javax.swing.*;      //使用 swing 元件請匯入 javax.swing.*套件
06
07 //MyJFrame(記憶大考驗遊戲視窗)繼承 JFrame 視窗元件
08 class MyJFrame extends JFrame  {
09    // 宣告 icons[0]~icons[4] 用來存放
10    // 0.jpg(? 問號圖), 1.jpg, 2.jpg, 3.jpg, 4.jpg 五個圖示
11    private ImageIcon[] icons = new ImageIcon[5];
12    // 宣告 jbtn[0]~jbtn[7] 八個按鈕
13    private JButton[] jbtn = new JButton[8];
14    // 宣告 jbtnOk 確定鈕, jbtnf 表示第一次按下的按鈕, jbtns 表示第二次按下的按鈕
15    private JButton jbtnOk, jbtnf, jbtns;
16    // 宣告 f 表示第一次按下按鈕取得的字串，s 表示第二次按下按鈕取得的字串
17    String f = "", s = "";
18    // 宣告 num 表示按下按鈕的次數；win 表示共猜對幾組圖示
19    int num = 0, win = 0;
20    // 建立 rnd[0]~rnd[7]用來存放記憶大考驗每張圖所代表的編號
21    int[] rnd = new int[8];
22    // MyJFrame 建構式
23    MyJFrame()
24    {
25       // 不使用版面配置
26       super.setLayout(null);
27       // 視窗標題設為 "記憶大考驗"
28       super.setTitle("記憶大考驗");
29       // 設定 icons[0]~icons[4]元件的圖示為 memberImg 資料夾下的 0.jpg~4.jpg
30       // 其中 0.jpg 為?問號圖
31       for(int i = 0; i < icons.length; i++) {
32          icons[i] = new ImageIcon
                 (".\\memoryImg\\" + String.valueOf(i) + ".jpg");
33       }
34
```

```
35        // 在視窗放入 jbtnOk 確定鈕
36        jbtnOk = new JButton("確定");
37        jbtnOk.setBounds(20, 260, 80, 30);
38        jbtnOk.setFont(new Font("微軟中黑體",Font.PLAIN, 18));
39        add(jbtnOk);
40        // 指定 jbtnOk 確定鈕的傾聽者為 ActionListener 匿名物件
41        // 按下確定鈕時會執行該物件的 actionPerformed 方法
42        jbtnOk.addActionListener(new ActionListener() {  //按下確定鈕執行此處
43           public void actionPerformed(ActionEvent evt) {
44              // 建立 ary[0]~ary[7] 用來存放圖示的編號，編號相同的為同一組
45              int[] ary = new int[]{1,1,2,2,3,3,4,4};
46              int n = 0;    // 用來存放產生的亂數
47              int max = ary.length - 1;
48              // 使用迴圈 jbtn[0]~jbtn[7]進行亂數存放 1.jpg~4.jpg
49              // 編號相同為同一組
50              for (int i=0; i<ary.length; i++) {
51                 n = (int)Math.round((Math.random() * max));
52                 rnd[i] = ary[n];
53                 ary[n] = ary[max];
54                 max--;
55                 jbtn[i].setActionCommand(String.valueOf(rnd[i]));
56                 jbtn[i].setToolTipText(String.valueOf(i));
57                 jbtn[i].setIcon(icons[0]);
58                 jbtn[i].setEnabled(true);
59              }
60           }
61        });
62
63        // 建立 jbtn[0]~jbtn[7] 八個按鈕，排成兩行，一行有四個按鈕
64        int x = 0, y = 0;
65        for (int i = 0; I < jbtn.length; i++){
66           jbtn[i] = new JButton();
67           jbtn[i].setBounds(x * 100 + 20, y * 120 + 20, 80, 100);
68           jbtn[i].setIcon(icons[0]);        // 按鈕預設顯示 ? 問號圖
69           jbtn[i].setEnabled(false);
70           x++;
71           if (i % 4 == 3) {
72              y++;
```

```
73              x = 0;
74          }
75          // 在視窗放入 jbtn[0]~jbtn[7] 八個按鈕
76          add(jbtn[i]);
77          // 指定 jbtn[0]~jbtn[7] 八個按鈕的傾聽者為 ActionListener 匿名物件
78          // 當按下 jbtn[0]~jbtn[7] 時會執行該物件的 actionPerformed 方法
79          jbtn[i].addActionListener(new ActionListener() {
80              public void actionPerformed(ActionEvent evt) {
81                  num++;              // 按下按鈕次數加 1
82                  if (num == 1) {     // 第一次按下
83                      // 取得第一次按下按鈕代表的字串
84                      f = evt.getActionCommand();
85                      // 取得第一次按下按鈕
86                      jbtnf = (JButton)evt.getSource();
87                      jbtn[Integer.parseInt(jbtnf.getToolTipText())]
                                    .setIcon(icons[Integer.parseInt(f)]);
88                  } else if (num == 2) {      // 第二次按下
89                      // 取得第二次按下按鈕代表的字串
90                      s = evt.getActionCommand();
91                      // 取得第二次按下按鈕
92                      jbtns= (JButton)evt.getSource();
93                      jbtn[Integer.parseInt(jbtns.getToolTipText())]
                                .setIcon(icons[Integer.parseInt(s)]);
94                      //若第一次按下按鈕的 f 字串與第二次按下按鈕的 s 字串相等
95                      //且第一次按下按鈕與第二次按下按鈕不是同一個，
                        //則表示猜對一組圖示
96                      if (f.equals(s) && jbtns!=jbtnf) {
97                        JOptionPane.showMessageDialog(null, "猜對了!");
98                        jbtnf.setEnabled(false);    //第一次按鈕的按鈕停用
99                        jbtns.setEnabled(false);    // 第二次按鈕的按鈕停用
100                       win++;                      // 猜對組數加一
101                       if (win == 4) {             // 若猜對四組
102                          JOptionPane.showMessageDialog
                                                (null, "全對了...ya!");
103                       }
104                     } else {
105                       //若沒有猜對任一組圖示，則之前按下的按鈕皆還原成?問號圖示
106                       JOptionPane.showMessageDialog(null, "不對哦!");
```

```
107                         jbtnf.setIcon(icons[0]);
108                         jbtns.setIcon(icons[0]);
109                     }
110                     f = "";
111                     s = "";
112                     num = 0;
113                 }
114             }
115         });
116     }
117
118     // 設定視窗大小為寬 430, 高 360
119     setSize(430, 360);
120     // 顯示視窗
121     setVisible(true);
122     // 設定按視窗的關閉鈕會結束程式
123     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
124   }
125 }
126
127 //主程式
128 public class Program {
129   public static void main(String[] args){
130     // 建立 MyJFrame 視窗(記憶大考驗遊戲)
131     new MyJFrame();// 執行第 8~125 行建立 MyJFrame 類別物件(記憶大考驗遊戲)
132   }
133 }
```