第七章 陣列

7-1 陣列結構

陣列（Array）是一群相同**『資料型態』**的變數整合而成的另一種變數，並使用一個共用的參考名稱。其中『資料型態』可以是一般基本資料型態（如 int、float、double...等），也可以是一個『物件』型態（如 String...等）。

如圖 7-1 所示（一維陣列結構），陣列 a 包含了 8 個相同資料型態（int）的變數，這些變數各自獨立的，也可分別儲存資料，之間並不相衝突；但為了使這些資料之間也某些連帶關係，利用相同的變數名稱（如 a），各自給予獨立的指標（a[i]，i = 0, …, 7）。

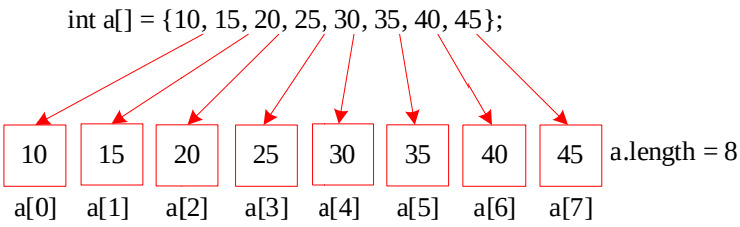

圖 7-1 一維陣列結構

對資料整合與運用（如資料結構）而言，陣列是非常重要的資料型態，可用來描述許多真實環境的現象。本章先介紹一些陣列的基本觀念與運用。

7-2 一維陣列處理

7-2-1 一維陣列宣告

『一維陣列』(One dimension array)表示僅有一行或一列的資料結構，但一列陣列可能包含若干個行；一行陣列也可能包含多個列之結構，如圖 7-1 所示。一維陣列僅需要一個指標變數(如 i)來索引陣列中哪一個元素（如 a[i]，i = **3**）。Java 語言宣告陣列的基本語法說明如下：

（A）陣列動態宣告：

其實在 Java 語言裡，無論哪一種基本資料型態，或以宣告完成的物件，它都是以物件的觀念來處理，尤其在陣列資料型態裡更是明顯。如同物件一樣，宣告某一陣列變數，只不過產生了資

料型態架構，必須再經由 new 敘述才能在主記憶體裡取得變數空間。基本語法如下：

陣列宣告的語法：	範例：
Data_type Array_name[]; Array_name = new Data_type[number];	int course[]; course = new int[20];
Data_type[] Array_name = new Data_type[num]; Array_name.length() = num;	int [] course =new int [20]; course.length() = 20;

第一個敘述（如 int course[]）功能是宣告某一變數（course）為整數（int）的陣列變數；第二個敘述（如 course = new int[20]）功能是由主記憶體取得 20 個整數空間，並配置給 course 變數使用。此兩條敘述執行完畢後，則產生 course[0]、course[1]...等、course[19]；需注意陣列指標是由 0 開始計算，如有 n 個元素，則最高指標是 n-1。 另外，系統也會自動產生 length 變數，紀錄該變數的長度如何，如 course = int[20]，則 course.length = 20。上述範例亦可簡寫成 int[] course = new int[20]，一行敘述句完成宣告。

（B）陣列宣告並給予初值

宣告陣列時可以給予初值，則表示由記憶體取得空間後立即存入初值數值，陣列的大小就如此被固定下來（length 變數內容），也不需再由 new 敘述來配置空間。語法格式如下：

陣列宣告並給予初值：	範例：
Data_type Array_name[] = { …};	int course[] = {89, 90, 60, 70, 80}; // **注 course.length = 5**

上述執行後，會產生 5 個元素，分別是 course[0] = 89、course[1] = 90、course[2] = 60、course[3] = 70 與 course[4] = 80，以及 course.length = 5。吾人還是取用更多範例說明如下：

陣列宣告範例：	說明：
float weight[]; weight = new float[30]	宣告 weight 陣列變數為浮點數型態。 產生 30 個浮點數元素；**weight.length=30**。
float weight = new float[30];	同上
String[] names; names = String[20];	宣告 names 陣列變數為字串型態。 產生 20 個字串元素；names.length=20。
char keys[] = {‘A’, ‘B’, ‘C’};	宣告產生字元陣列，並給初值。

7-2-2 範例研討：印出股票歷史價

（A）程式功能：Ex7_1.java

吾人利用陣列 course[] = {78.8, 72.3, 61, 56, 87, 66.3, 74.5, 88, 76, 58}; 儲存某一支股票最近 10 個交易日的收盤價，請列印出其內容；期望操作介面如下：

```
D:\Java1_book\chap7>java Ex7_1
78.80   72.30   61.00   56.00   87.00   66.30   74.50   88.00   76.00   58.00
```

（B）製作技巧研討：

陣列中每一個元素都是獨立變數，必須利用指標變數一個接一個取出再列印。本範例 course[] 包含了 10（= course.length）個元素，則利用指標 i = 0, 1, …, 9，分別索引每一元素 course[i] 的內容。譬如，當 count =0，則 course[0] = 78.8；count=1，則 course[1] = 72.3，依此類推。因此，吾人可利用 for 迴圈，條件初始值為 i = 0、判斷條件 count < course.length、增減量為 i++。

（C）程式範例：

```
01  //Ex7_1.java
02
03  public class Ex7_1 {
04      public static void main(String args[]) {
05          double course[]={78.8, 72.3, 61, 56, 87,
06                           66.3, 74.5, 88, 76, 58};
07
08          /* 由 course[0]~ course[course.length-1] 列印陣列 */
09          for(int i=0; i<course.length; i++)
10              System.out.printf("%.2f   ", course[i]);
11          System.out.printf("\n");              // 列印完畢，換行
12      }
13  }
14
```

（D） 程式重點分析：

(1) 第 5~6 行：『**double course[] = { …}**』。宣告 course 陣列變數為 double 資料型態，也立即給予初值。

(2) 第 9~10 行：『**for(..; i<course.length; ..) System.out.printf("%.2f   ", course[i];**』。其中 length 變數表示元素的個數（ 10），但陣列元素是由 0 開始計算，因此最後一個元素是

length -1。另外，列印格式是 "%.2f "，其表示印出有 2 位小數點的浮點格式（%.2f），後面再接兩個空白格。

7-2-3 範例研討：最近十天的平均價

（A）程式功能：Ex7_2.java

請修改 Ex7_1.java 程式，使其功能不但輸出最近 10 個交易日的收盤價，並計算出他的平均價如何。【利用陣列 course[] = {78.8, 72.3, 61, 56, 87, 66.3, 74.5, 88, 76, 58}; 儲存某一支股票最近 10 個交易日的收盤價】，期望操作介面如下：

```
78.80  72.30  61.00  56.00  87.00  66.30  74.50  88.00  76.00  58.00
最近 10 天的平均價 = 71.79
```

（B）程式範例：

```
1   //Ex7_2.java
2   /* 吾人利用陣列 course[]={78.8,72.3,61,56,87,66.3,74.5,88.76.58}；
3
4   * 儲存某一支股票最近10個交易日的收盤價，請計算它的平均價 */
5
6   public class Ex7_2 {
7       public static void main(String args[]) {
8               double course[]={78.8, 72.3, 61, 56, 87,
9                                       66.3, 74.5, 88, 76, 58};
10              double sum=0, ave;
11              for(int i=0; i<course.length; i++) {
12               System.out.printf("%.2f  ", course[i]);
13               sum = sum + course[i];
14              }
15              System.out.printf("\n");
16              ave = sum /course.length;
17              System.out.printf("最近 10 天的平均價 = %.2f", ave);
18      }
    }
```

7-2-4 自我挑戰：最近十天的最高/最低價

（A）程式功能：PM7_1.java

請修改 Ex7_1.java 程式，使其功能不但輸出最近 10 個交易日的收盤價，並比較輸出其中最

高與最低價格如何。【利用陣列 course[] = {78.8, 72.3, 61, 56, 87, 66.3, 74.5, 88, 76, 58}; 儲存某一支股票最近 10 個交易日的收盤價】，期望操作介面如下：

```
78.80   72.30   61.00   56.00   87.00   66.30   74.50   88.00   76.00   58.00
最近 10 天的最高價格 = 88.00
最近 10 天的最低價格 = 56.00
```

（B）程式設計技巧：(程式片段)

```
1  public class PM7_1 {
2      public static void main(String args[]) {
3  // 這是程式片段
4              double Max=0.0, Min=9999.9;
5              for(int i=0; i<course.length; i++) {
6               System.out.printf("%.2f    ", course[i]);
7               if (Max < course[i])
8                     Max = course[i];
9               if (Min > course[i])
10                    Min = course[i];
11            }
12 ……// 這是程式片段
13     }
14 }
```

7-2-5 範例研討：記錄最近 30 天的收盤價

（A）程式功能：Ex7_3.java

如想隨時得到某股票的歷史平均收盤價，必須紀錄該股票每日收盤價格；一日接一日的交替變化，需要紀錄的資料也可能無限延伸。當然不可能無限紀錄該股票的收盤價，再說太久之前的資料也可能沒有任何意思。但歷史資料記錄太少的話，也可能影響評估股價的正確性。本範例需要 30 日之前股票的收盤價，可利用由一維陣列製作的『佇列器』(Queue) 來紀錄。圖 7-1 利用一只 30 個元素的一維陣列，儲存台積電近 30 日收盤價 (假設數值)；stock[0] 為前一日、stock[1] 為前二日、依此類推，stock[29] 記錄前第 30 日股價。當欲新登錄資料，則所有陣列元素往前移一個位置，原 stock[29] 資料拋棄，新資料填入 stock[0]。

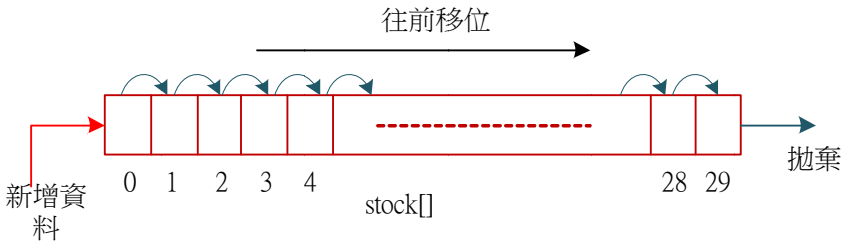

圖 7-2 紀錄過去 30 日收盤價

假設某位分析師利用陣列 stock[] = {78, 72, 61, 56, 87, 66, 74, 88, 76, 58, 65, 57, 90, 78, 67, 89, 56, 77, 56, 87, 67, 80, 77, 86, 67, 75, 86, 98, 88, 78}; 儲存台積電最近 30 個交易日的收盤價，其中 stock[0] 為昨日股價，又 stock[29] 是之前第 30 日的收盤價。

我們必須設計一套程式，除了可以隨時顯示最近 30 天的收盤價之外，也可以輸入每天的收盤價如何。注意，每天輸入收盤價之後，整個陣列會隨著移動，它會拋棄 30 天前收盤價，並擠入最近的收盤價（如圖 7-1），我們希望操作環境如下：

```
== 歡迎光臨 股票走勢分析系統 ==
(1) 列印 30 日歷史收盤價
(2) 登錄當日收盤價
(3) 離開系統
        請輸入工作選項 =>1

== 列印最近 30 日股價==
78.00   72.00   61.00   56.00   87.00
66.00   74.00   88.00   76.00   58.00
65.00   57.00   90.00   78.00   67.00
89.00   56.00   77.00   56.00   87.00
67.00   80.00   77.00   86.00   67.00
75.00   86.00   98.00   88.00   78.00

== 歡迎光臨 股票走勢分析系統 ==
(1) 列印 30 日歷史收盤價
(2) 登錄當日收盤價
(3) 離開系統
        請輸入工作選項 =>2
請登錄當日收盤股價 =>90
== 歡迎光臨 股票走勢分析系統 ==
(1) 列印 30 日歷史收盤價
(2) 登錄當日收盤價
```

```
(3) 離開系統
        請輸入工作選項 =>1

== 列印最近 30 日股價==
90.00   78.00   72.00   61.00   56.00
87.00   66.00   74.00   88.00   76.00
58.00   65.00   57.00   90.00   78.00
67.00   89.00   56.00   77.00   56.00
87.00   67.00   80.00   77.00   86.00
67.00   75.00   86.00   98.00   88.00

== 歡迎光臨 股票走勢分析系統 ==
(1) 列印 30 日歷史收盤價
(2) 登錄當日收盤價
(3) 離開系統
        請輸入工作選項 =>
```

（B）程式範例

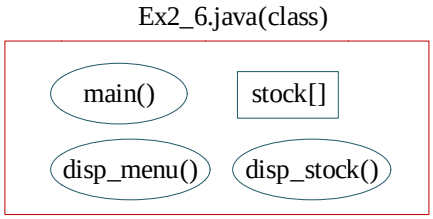

圖 7-3

程式範例如下：

```
01  import java.util.*;
02  public class Ex2_6{
03     static double stock[]={78, 72, 61, 56, 87,
04                            66, 74, 88, 76, 58,
05                            65, 57, 90, 78, 67,
06                            89, 56, 77, 56, 87,
07                            67, 80, 77, 86, 67,
08                            75, 86, 98, 88, 78};
09     public static void main(String args[]) {
10          Scanner keyin = new Scanner(System.in);
11          double cost;
12          int select;
13          disp_menu();
14          select = keyin.nextInt();
15          while(select != 3) {
16               switch(select) {
```

```
17                  case 1:
18                      disp_stock();
19                      break;
20                  case 2:
21                      System.out.printf("請登錄當日收盤股價 =>");
22                      cost = keyin.nextDouble();
23                      for (int i=(30-1); i>=1; i--)
24                          stock[i] = stock[i-1];
25                      stock[0] = cost;
26                      break;
27                  default:
28                      System.out.printf("輸入錯誤 !! 請重新輸入\n");
29              }
30              disp_menu();
31              select = keyin.nextInt();
32          }
33      }
34      static void disp_menu() {
35          System.out.printf("== 歡迎光臨 股票走勢分析系統 ==\n");
36          System.out.printf("(1) 列印 30 日歷史收盤價\n");
37          System.out.printf("(2) 登錄當日收盤價\n");
38          System.out.printf("(3) 離開系統\n");
39          System.out.printf("\t 請輸入工作選項 =>");
40      }
41      static void disp_stock() {                  /* 列印 30 日內股價 */
42          System.out.printf("\n== 列印最近 30 日股價==\n");
43          for(int i=0; i<stock.length; i++) {
44              System.out.printf("%.2f   ", stock[i]);
45              if((i+1) % 5 == 0)
46                  System.out.printf("\n");       //列印五筆, 換行
47          }
48          System.out.printf("\n");                //列印完畢, 換行
49      }
50  }
```

7-2-6 自我挑戰：股票走勢分析系統

（A）程式功能：PM7_2.java

當股票分析師目標設定哪一股票（如台積電）後，大多會紀錄該股票過去收盤價，再利用資

料分析最近該股的走勢，如果今天收盤價高於 15、30 日平均價，則稱為『多頭』走勢；如果高於 15 日但低於 30 日平均線，稱之為『盤整轉向多頭』；低於 15 日又高於 30 日，則稱為『盤整轉向空頭』；如果低於兩者，則為『空頭』走向。假設某位分析師利用陣列 stock[] = {78, 72, 61, 56, 87, 66, 74, 88, 76, 58, 65, 57, 90, 78, 67, 89, 56, 77, 56, 87, 67, 80, 77, 86, 67, 75, 86, 98, 88, 78}; 儲存台積電最近 30 個交易日的收盤價，其中 stock[0] 為昨日股價，又 stock[29] 是之前第 30 日的收盤價。請製作一股票走勢分析系統，且允許分析師增加當天收盤價、列印歷史收盤價、股票走勢如何；期望操作介面如下：

(1) 期望操作介面有 4 個功能選項

```
D:\Java2_book\chap2>java PM2_4
== 歡迎光臨 股票走勢分析系統 ==
(1) 列印 30 日歷史收盤價
(2) 登錄當日收盤價
(3) 分析目前股票走勢
(4) 離開系統
        請輸入工作選項 =>
```

(2)選擇列印 30 日歷史收盤價(選擇 1)操作如下：

```
        請輸入工作選項 =>1

== 列印最近 30 日股價==
78.00   72.00   61.00   56.00   87.00
66.00   74.00   88.00   76.00   58.00
65.00   57.00   90.00   78.00   67.00
89.00   56.00   77.00   56.00   87.00
67.00   80.00   77.00   86.00   67.00
75.0  86.00   98.00   88.00   78.00
```

(3)選擇登錄當日收盤價(選擇 2)操作如下：

```
        請輸入工作選項 =>2
請登錄當日收盤股價 =>90
```

(4)選擇分析目前股價走勢(選擇 3)操作如下：(空頭走勢)

```
        請輸入工作選項 =>3
```

```
請輸入目前股價 =>50
15 日平均= 73.07, 30 日= 75.07
目前是空頭走勢
```

(5)選擇分析目前股價走勢(選擇 3)操作如下：(多頭走勢)

```
請輸入工作選項 =>3
請輸入目前股價 =>98
15 日平均= 73.07, 30 日= 75.07
目前是多頭走勢
```

(6)選擇分析目前股價走勢(選擇 3)操作如下：(盤整走勢)

```
請輸入工作選項 =>3
請輸入目前股價 =>75
15 日平均= 73.07, 30 日= 75.07
目前是盤整轉向多頭走勢
```

（B）製作技巧研討：

如何記錄過去 30 天交易日的收盤價，如圖 2-# 與範例 Ex2_6 分別表示製作方法。另外，計算 15 日平均價，則計算 stock[0] 到 stock[14] 的平均值；30 日平均價是 stock[0]～stock[29] 平均值，如此就可判斷該股票的走勢如何。如果目前價格來分析：

(1) 大於 15、30 日平均價，則為**『多頭』走勢**；

(2) 如低於 15、30 日兩者的平均價，則為**『空頭』走勢**；

(3) 如高於 15 日且低於 30 日是**『盤整轉向多頭』**；

(4) 其他（低於 15 日但高於 30）是**『盤整轉向空頭』**。

另一個重點，列印所有資料時，為了所輸出的報表較漂亮，每列印五筆資料後，就需換行。我們就依照該系統所應有的功能，將它分別寫成四個方法，其中 main() 為主程式、disp_menu() 功能是顯示功能選單、cal_ave(k) 為計算股票平均價使用，k 為計算最近天數 (15 或 30 日)、disp_stack() 為顯示所有股價的程式，整個類別架構如下圖所示。

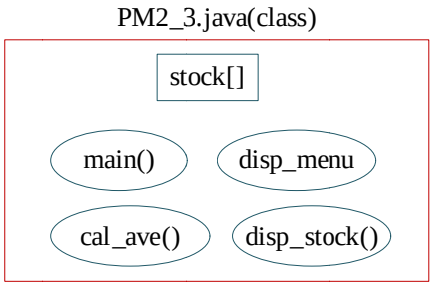

圖 7-4

虛擬碼提示如下：

```
導入輸入套件（import java.util.*;）;

主類別範圍 {

    宣告靜態陣列並設定初值（static double stock[]={.....}；

    主程式範圍 {

        宣告相關物件與變數；

        顯示功能選項（disp_menu() ）;

        讀取功能選項（select = keyin.nextInt() ）;

        while (select !=4) {
            switch(select) {

                case 1：顯示 30 日收盤價（disp_stock() ）；

                case 2：讀取當日收盤價（cost ）;

                        移位陣列（for (int I= 29 ~ 1) stock[I] = stock[I-1]; ）;

                        當日填入（stock[0] = cost; ）;

                case 3：讀取目前股價（cost ）;

                        計算 15 日平均價（ave_15 = cal_ave(15) ）;

                        計算 30 日平均價（ave_30 = cal_ave(30) ）;

                        顯示平均價；

                        if ((cost>ave_15) && (cost>ave_30))

                            顯示多頭走勢；

                        else if   ((cost <ave_15) && (cost <ave_30)

                            顯示空頭走勢；

                        else if ((cost >ave_15) && (cost <ave_30))

                            顯示盤整轉向多頭走勢；

                        if ((cost>ave_15) && (cost>ave_30))

                            顯示盤整轉向空頭走勢；
```

```
33                  } // switch/case 結束
34
35                  顯示功能選項（disp_menu()）;
36
37                  讀取功能選項（select = keyin.nextInt()）;
38
39              } // while 結束
40
41          } // 主方法結束
42
43          顯示選單方法範圍（static void disp_menu()）{.....}
44
45          計算 k 日平均價範圍（static double cal_ave(int k)){
46              double sum=0;
                for(int I=0; I<k; I++)
                    sum = sum + stock[I];
                return sum/k;
            } // cal_ave() 結束

            列印 30 日內所有股價範圍（static void disp_stock()）{
                for(int i=0; i<stock.length; i++) {
                    System.out.printf("%.2f  ", stock[i]);
                    if(( i+1) % 5 == 0)
                        System.out.printf("\n");      //列印五筆, 換行
                }
                System.out.printf("\n");              // 列印完畢, 換行
            } // disp_stock 結束
        } // 主類別結束
```

7-2-7 自我挑戰：印製國字收據

(A)程式功能：PM7_3.java

請製作『真自在遊民收容所』的捐款收據，功能是系統允許輸入捐款人姓名與金額，之後印出收據但捐款數字需由國字印出，期望操作介面如下：

```
D:\Java1_book\chap7>java PM7_3
請輸入大德先生/小姐姓名 =>無名氏
請輸入捐款金額 =>3212345
列印收據如下：

**** 真自在遊民收容所      捐款收據 ****

      感謝 無名氏 先生/小姐大德贊助
```

```
                捐款 3212345 元整

總計 = 零 仟 參 佰 貳 拾 壹 萬 貳 仟 參 佰 肆 拾 伍 元整
```

（B）製作技巧提示：

本範例較困難的地方是數字轉換國字後，如何寫入適當的位置。我們設定一個範圍，比較容易設計列印格式，因此假設捐款最高為 9 千萬餘元（99999999，超過則另開收據）；圖 7-2 為本範例列印格式（大多如此）。另外，我們可以利用整數相除得到某一位數（...百位、十位、個位），再利用餘數運算，得到取出某一位數後的剩餘值；另選一個陣列(int num[])存放每一位數的數字。讀取輸入捐款金額後（value），除以 1 千萬（10000000）得到千萬位的數字，再利用相同的數經過餘數運算，則扣除千萬位數，得到百萬以下的數值，依此類推則可得到相對位數的數字，並依序存入 num[] 陣列內。同時，選用 chinese[] 陣列存放每種數字的相對應國字，chinese[0] 存『零』（0）、chinese[1] 存『壹』(1)...等依此類推。

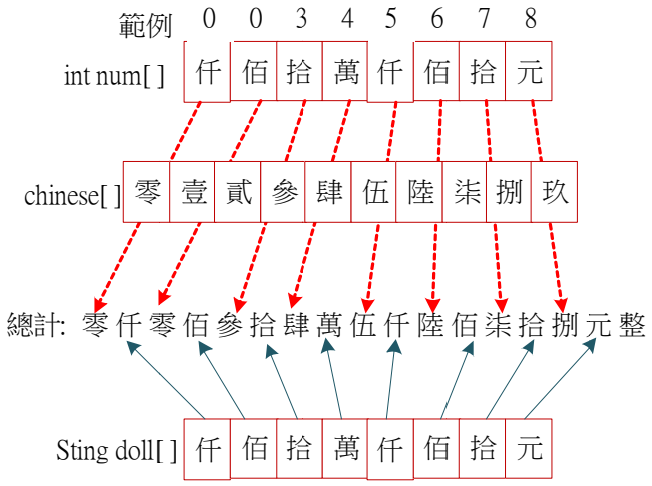

圖 7-5 國字列印格式

吾人再選用 doll[] 陣列存放列印次序的相對位數文字；則利用 chinese[num[i]]與 doll[i] 陣列交互列印，其中 i 為列印位數的指標；如此即可得到國字列印，虛擬碼提示如下：

```
01 宣告國字數字陣列（String chinese[] = {"零", "壹", "貳", "參", .., "玖"}）;
02
03 宣告列印格式陣列（String doll[] = {"仟", "佰", "拾","萬", “仟”, “佰”, “拾”, "元"}；
04 宣告位數數字陣列（int[] num = new int[9] ）;
05
06 宣告最高數字變數（千萬，base=10000000 ）;
07 讀取捐款姓名（name ）;
08
```

```
09  讀取捐款額（value）;
10
11  備份捐款額（value1 = value）;
12  取出各個位數的數字 :（最高 8 位數）
13
14          for (int i=0; i<8; i++) {        //i=0, 仟萬; i=1, 佰萬; i=2, 拾萬; …
15              num[i] = value / base;
16              value = value % base;
17              base = base /10;
18          }
19  列印捐款人姓名及金額（name, value1）;
    列印捐款的國字 :（最高 8 位數）
          System.out.printf("\n 總計 =");
          for(int i=0; i<8; i++)
              System.out.printf(" %s %s", chinese[num[i]], doll[i]);
          System.out.printf("整 \n");
```

7-3 二維陣列處理

既然陣列是由若干個相同資料型態的變數整合而成，變數的排列格式也延伸了不同的陣列格式。如果所有變數排列成一行或一列，則稱之為『一維陣列』(如 array[])；如果排列成平面形狀，則稱為『二維陣列』(如 array[][])；如果是立體形狀，則為『三維陣列』(如 array[][][])；如是四度空間型態，則是『四維陣列』(如 array[][][][])，其中每一維都需要一個變數作為指標，索引其相對位置。各種陣列型態都有其運用範圍，如沒有特殊情況的話，還是一維與二維陣列的應用最普遍，本書僅介紹到二維陣列，更多維數陣列的處理方式，也大致相同的。

7-3-1 二維陣列宣告

將變數排列成『縱橫』的平面形狀，則稱之為『二維陣列』，如圖 7-3 所示。二維陣列需要兩個位置指標，如 score[x][y] 變數，前面指標 (如 x) 是標示第幾行數 (由 0 開始)； 第二個指標 (如 y)，則標示第幾列數；如是 4 * 3 陣列共計有 12 個相同型態 (int) 變數的元素。另外，陣列被宣告完成之後 (如 int score[][] = { ..})，則 score.length 變數儲存該陣列的行數；又 score[x].length 儲存該行 (x) 元素的數目 (即是列數目)。宣告語法如下所示：

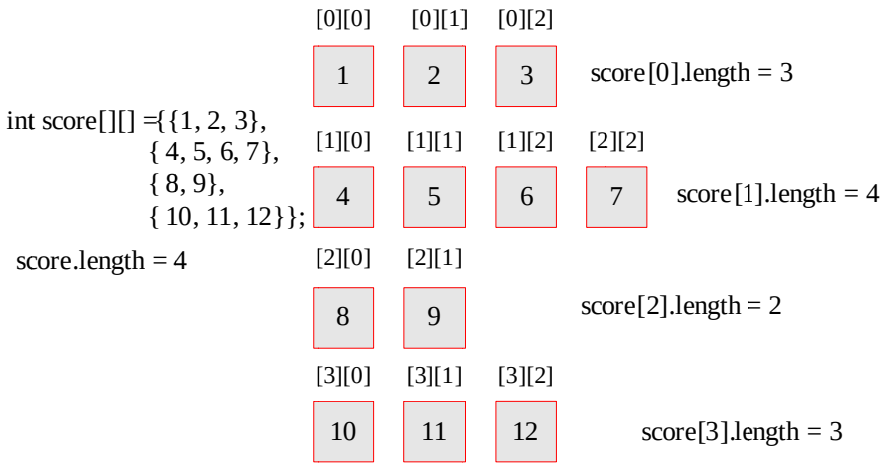

圖 7-6 二維陣列的結構

二維陣列宣告語法：	範例：

Data_type Array_name[][];
Array_name = new Data_type[num1][num2];
Data_type[][] name = new Data_type[num1][num2];
	int score[];
	score = int[20][10];
	int[][] score = new int [20][10];

Data_type Array_name[] []= { …};	int score[][] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {10, 11, 12}};

第一行語法是宣告產生二維陣列，第二行則再給予初值。如敘述句如：int [] [] score = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {10, 11, 12}};則產生各個元素與變數內容如下：

score[0][0] = 1, score[0][1]=2, score[0][2]=3 =>score[0].length=3
score[1][0] = 4, score[1][1]=5, score[1][2]=6 =>score[1].length=3
score[2][0] = 7, score[2][1]=8, score[2][2]=9 =>score[2].length=3
score[3][0] = 10, score[3][1]=11, score[3][2]=12 =>score[3].length=3
score.length = 4

7-3-2 範例研討：印出二維陣列內容

（A）程式功能：Ex7_4.java

請編寫一程式，印出陣列 score [] [] = {{1, 2, 3}, {4, 5, 6},{7, 8, 9}, {10, 11, 12}} 內容並依照相對位置排列（平面形狀）。 期望操作介面如下：

```
D:\Java2_book\chap3>javac Ex3_1.java

D:\Java2_book\chap3>java Ex3_1
        1       2       3
```

```
   4    5    6
   7    8    8
  10   11   12
```

（B）製作技巧研討：

吾人利用 x 與 y 變數作為陣列位置索引，x 是行數指標，y 為列數指標，則任何一元素可由 score[x][y] 表示。如欲平面型態印出陣列內容，則需要二重迴圈敘述，外迴圈標示共計列印幾行（x = 0, 1, …, score.length）；內迴圈索引每行的列數（y = 0, 1, 2, …, score[x].length）。

（C）程式範例：

```
01 //Ex7_2.java
02
03 public class Ex7_2 {
04       public static void main(String args[]) {
05             int score[][]={{1, 2, 3},
06                            {4, 5, 6},
07                            {7, 8, 8},
08                            {10, 11, 12}};
09             int x;                          // 行指標 (外迴圈)
10
11             int y;                          // 列指標 (內迴圈)
12             for(x=0; x<score.length; x++) {
13                   for(y=0; y<score[x].length; y++)
14                         System.out.printf("\t%2d", score[x][y]);
15                   System.out.printf("\n");
16             }
17       }
   }
```

（D） 程式重點說明：

(1) 第 5~8 行：『**int score[][] = { …};**』。宣告二維陣列並給予初值。

(2) 第 11~15 行:『**for(x=0; x<score.length; x++) {…}**』。外迴圈指定列印行數，其中 score.length 表示該陣列的行數。

(3) 第 12~13 行：『**for(y=0; y<score[x].length; y++){…}**』。內迴圈指定每行的列的數目，其中 score[x].length 表示該行（x）列的數目。

(4) 第 13 行：『**System.out.printf("\t%2d", score[x][y]);**』。列印 score[x][y] 元素的內容，列印格式是開始先跳一個『Tab』（\t）定格，再以兩個位置的格式列印整數（%2d）。

(5) 第 14 行：『**System.out.printf("\n");**』。每行列印完成之後，跳至下一行。

7-3-3 自我挑戰：二維陣列內容加倍

（A）程式功能：PM7_4.java

請編寫一程式，請修改 Ex7_2 範例，將陣列 score [][] = {{1, 2, 3}, {4, 5, 6},{7, 8, 9}, {10, 11, 12}} 內容加 3 倍，再依照相對位置排列（平面形狀）。 期望操作介面如下：

```
D:\Java2_book\chap3>java Ex3_2
原陣列內容：
        1       2       3
        4       5       6
        7       8       8
       10      11      12
加倍後陣列內容：
        2       4       6
        8      10      12
       14      16      16
       20      22      24
```

（B）製作技巧提示：

我們建構一個比較容易擴充的程式，將 score[][] 宣告成類別變數，使其允許類別內所有方法使用，又將列印陣列的功能製作成一個獨立的 disp_arr() 方法，程式架構如圖 3-2 所示。

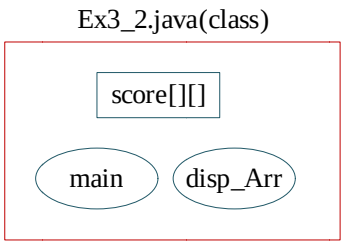

圖 7-7

(C) 程式片段

```
01  //Ex3_2.java
02  public class Ex3_2 {
03     /*宣告類別變數 score[] */
04
05     static int score[][]={{1, 2, 3},
06                           {4, 5, 6},
07                           {7, 8, 8},
```

```
08                                  {10, 11, 12}};
09      /* 主程式 main() */
10      public static void main(String args[]) {
11
12          System.out.printf("原陣列內容：\n");
13          disp_Arr();
14          for(int x=0; x<score.length; x++) {          // 行指標 (外迴圈)
15
16              for(int y=0; y<score[x].length; y++)     // 列指標 (內迴圈)
17                  score[x][y] = score[x][y] * 2;
18          }
19          System.out.printf("加倍後陣列內容：\n");
20          disp_Arr();
21      }
22
23      /* 列印陣列方法 disp_arr()  */
24      static void disp_Arr() {
25          for(int x=0; x<score.length; x++) {
26              for(int y=0; y<score[x].length; y++)
27                  System.out.printf("\t%2d", score[x][y]);
28          System.out.printf("\n");
            }
      }
}
```

7-3-4 範例研討：印出二維陣列的轉陣列（S^{-T}）

（A）程式功能：Ex7_5.java

請編寫一程式，請將陣列 score [][] = {{1, 2, 3}, {4, 5, 6},{7, 8, 9}, {10, 11, 12}} 轉移（S^{-T}），再依照相對位置排列（平面形狀）。 期望操作介面如下：

```
D:\Java1_book\chap7>java Ex7_5
列印原陣列如下：
        1       2       3
        4       5       6
        7       8       9

列印轉移陣列如下：
        1       4       7
        2       5       8
        3       6       9
```

(B) 製作技巧

我們只要將由陣列內容取出的次序轉換就可以，原陣列取出是 score[i][j]，轉移陣列取出是次序是 score[j][i]。當外迴圈 i=0 時，印出順序 (j=0、1,、2) 是 score[0][0]=1、score[1][0]=4、score[2][0]=7，當 i=1 時，印出順序是 score[0][1]=2、score[1][1]=5、score[2][1]=8，當 i=2 時，印出順序是 score[0][2]=3、score[1][2]=6、score[2][2]=9。

(C) 程式範例

```
01 public class Ex7_5{
02     public static void main(String args[]) {
03         int score[][] = {{1, 2, 3},
04                          {4, 5, 6},
05                           7, 8, 9}};
06
07         System.out.printf("列印原陣列如下：\n");
08         for(int i=0; i<score.length; i++){
09             for(int j=0; j<score[i].length; j++)
10                 System.out.printf("\t%2d", score[i][j]);
11             System.out.printf("\n");
12         }
13         System.out.printf("\n列印轉移陣列如下：\n");
14         for(int i=0; i<score.length; i++){
15             for(int j=0; j<score[i].length; j++)
16                 System.out.printf("\t%2d", score[j][i]);
17             System.out.printf("\n");
18         }
19     }
20 }
```

7-3-5 自我挑戰：列印直式唐詩

（A）程式功能：PM7_5.java

請製作一套唐詩列印系統，功能是能將所將輸入的唐詩(五言四句)，分別以橫式與直式印出。期望操作介面如下：

```
G:\Examples\chap7>java PM7_5
==  五言四句唐詩  列印  ==
請輸入第  1  句(五個字) =>山中相送罷
請輸入第  2  句(五個字) =>日幕掩柴扉
```

```
請輸入第 3 句(五個字) =>春草明年綠
請輸入第 4 句(五個字) =>王孫歸不歸

== 橫式列印唐詩 ==
山 中 相 送 罷
日 幕 掩 柒 扉
春 草 明 年 綠
王 孫 歸 不 歸

== 直式列印唐詩 ==
山 日 春 王
中 幕 草 孫
相 掩 明 歸
送 柒 年 不
罷 扉 綠 歸
```

（B）製作技巧提示：

本系統需要二維陣列（poem[][]; ），將詩句裡每一個文字儲存於陣列的元素裡，才可分別以橫式或直式印出。其中包含兩困難點，一者是如何將一串文字輸入，分別取出每一個文字，再存入陣列元素內；另一者是如何轉換陣列的行與列印出（直式列印）。前者，吾人將一串文字直接讀入系統並存放某字串變數（data）內（利用 Scanner 物件），再將該變數宣告成掃瞄物件（Scanner 物件）並指定分隔符號為沒有空格（useDelimiter("")），接著即可掃瞄取出每一個文字，再分別存入陣列內，如圖 7-4 所示。

(a) 橫式列印

	00	01	02	03	04
00	山	中	相	送	罷
10	日	幕	掩	柒	扉
20	春	草	明	年	綠
30	王	孫	歸	不	歸

(b) 直式列印

	00	10	20	30
00	山	日	春	王
01	中	幕	草	孫
02	相	掩	明	歸
03	送	柒	年	不
04	罷	扉	綠	歸

圖 7-8 poem[][] 陣列儲存內容

唐詩內所有文字都依序存入陣列後。原先是依照一行接一行填入文字，如果一行接一行印出，則如同輸入格式一樣是橫式輸出；如果將行與列倒過來印，則是直式輸出。虛擬碼提示如下：

```
導入相關套件（java.util.Scanner）;
宣告唐詩陣列（String[][] poem = new String[4][5] ）;
讀入五言四句唐詩：
        for(int i=0; i<4; i++) {
            System.out.printf("請輸入第 %d 句(五個字) =>", i+1);
            data = keyin.next();
            Scanner s = new Scanner(data).useDelimiter("");
            for(int j=0; j<5; j++)          // 分別讀取詩句中每一個文字
                poem[i][j] = s.next();
        }
列印橫式唐詩：
        for(int i=0; i<4; i++) {              // 列印 4 行
            for(int j=0; j<5; j++)            // 每行 5 個字
                System.out.printf("%s  ", poem[i][j]);
            System.out.printf("\n");
        }
列印直式唐詩：
        for(int j=0; j<5; j++) {              // 列印 5 行
            for(int i=0; i<4; i++)            // 每行 4 個字
                System.out.printf("%s  ", poem[i][j]);
            System.out.printf("\n");
        }
```

7-4 陣列線性搜尋

7-4-1 線性搜尋演算法

從陣列內尋找某一筆資料，最簡單的方法是線性搜尋法；演算法是由陣列的起頭開始比較尋找(比較內容)，如搜尋到則立即停止，否則繼續往下一個元素尋找，一直到陣列結束為止，如圖 7-5 所示。線性演算法所處理的陣列不需要特殊處理（如由大到小排列），最佳狀況是第一筆資料就找

到；最差狀態是最後一筆資料才搜尋到。

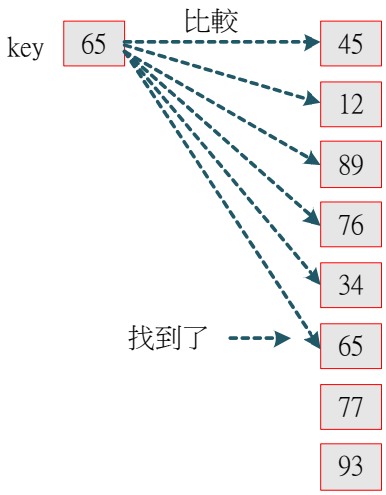

圖 7-9 線性搜尋法的運作

7-4-2 範例研討：實現線性搜尋法

(A)程式功能：Ex7_6.java

吾人利用 num[] 陣列儲存 10 個整數，輸入一個整數來查詢是否在 num 內，再顯示其執行結果，如下：

```
num[] = 20    13    45    24    42    34    22    89    19    70
請輸入一個數值 =>17
17 不在 num 陣列內
num[] = 20    13    45    24    42    34    22    89    19    70
請輸入一個數值 =>24
24 找到了
```

(B)程式範例：

```
01 import java.util.Scanner;
02 public class Ex7_6 {
03     public static void main(String args[]) {
04         Scanner keyin = new Scanner(System.in);
05             int value, flag=0, i;
06             int[] num = {20, 13, 45, 24, 42, 34, 22, 89, 19, 70};
07             System.out.printf("num[] = ");
08             for (int k=0; k<10; k++)
09                 System.out.printf("%d    ", num[k]);
               System.out.printf("\n");
```

```
10                System.out.printf("請輸入一個數值 =>");
11                value = keyin.nextInt();
12                i=0;
13                while (i < 10) {
14                      if (value == num[i]){
15                            flag = 1;
16                            break;
17                      }
18                      i++;
19                }
20                if(flag == 1)
21                        System.out.printf("num[%d] = %d 找到了\n", i, value);
22                else
23                 System.out.printf("%d 不在 num 陣列內", value);
24          }
25    }
26
```

7-4-3 範例研討：大樂透電腦選號

（A）程式功能：Ex7_7.java

請製作大樂透的電腦選號系統，系統能自動選出 6 個由 01～49 號碼，但這六個號碼都不可以重複。期望操作介面如下：

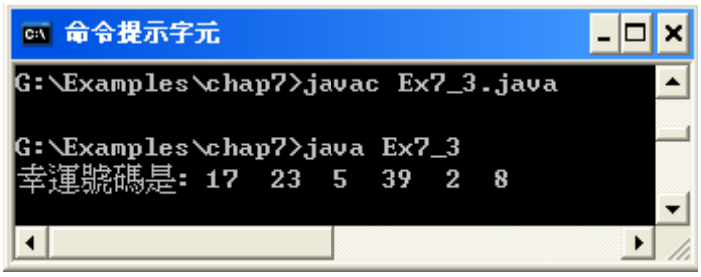

圖 7-10

（B）製作技巧研討：

編寫一個產生 6 個隨機亂數的程式並不困難，但如果要這 6 個亂數不重複的話，不加點小技巧是不行的。首先準備一個 6 個元素的陣列（int[] num = new int[6];），每次隨機選出號碼後，立即比較陣列內是否有重複號碼(之前所取的)，如果沒有則加入該陣列內。第 1 次選出號碼(i=0)，陣列還是空的(j=0)，則幾乎不用比對；如已選出 3 個號碼，下一個選出號碼就必須比對之前 3 個號碼是否有重複，依此類推。最後再印出以選出的 6 個號碼。

（B） 程式範例：

```
01  //Ex7_7.java
02
03  import java.lang.Math;
04  public class Ex7_7 {
05      public static void main(String args[]) {
06          int value, flag;                    // 隨機選取號碼
07
08          int[] num = new int[6];         // 儲存選取號碼
09          int i=0;
10          while (i < 6) {
11              flag=0;
12              value = 1 + (int)(Math.random()*46);
13              for (int j=0; j<i; j++) {           // 檢視已選出的號碼
14
15                  if (value == num[j]) {      // 是否重複
16                      flag = 1;               // 重複則放棄重來
17                      break;
18                  }
19              }
20              if (flag == 0) {
21
22                  num[i] = value;             // 沒重複則填入陣列
23                  i = i+1;
24              }
25          }
26          System.out.printf("幸運號碼是: ");
27          for (i=0; i<6; i++)
28              System.out.printf("%d    ", num[i]);
29          System.out.printf("\n");
        }
    }
```

（D）程式重點分析

- 第 9~16 行：『while{i<6} {…}』。選出 6 個不重複號碼。
- 第 11~17 行：『for(int j=0; j<i; j++) { if(value == num[j]) …}』。線性搜尋敘述句。
- 第 13~14 行：『if(value == num[j]) continue;』。如發現號碼重複，則放棄重來。

7-4-4 自我挑戰：最高與最低成績者

（A）程式功能：PM7_6.java

數學老師利用一個二維陣列儲存某一班級學生的成績，score[][] = {{411101, 70}, {411102, 80}, {411103, 75}, {411104, 90}, {411105, 85}, {411106, 65}, {411107, 83}, {411108, 78}}。請編寫一程式列印出該班成績最高與最低分數與姓名，期望操作介面如下：

```
D:\Java2_book\chap3>javac PM7_6.java

D:\Java2_book\chap3>java PM7_6
最高者  411104  成績= 90
最低者  411106  成績= 65
```

（B）製作技巧提示：

二維陣列(score[][])的每行(score[i], i =0, 1, 2 .., 7)紀錄一筆學生資料，第 0 列(score[i][0])儲存學號、第 1 列（score[i][1]）儲存數學成績。吾人可利用兩只一維陣列（max[] 與 min[]）儲存最高成績與最低成績，其中第 0 列（如 max[0]）儲存學號、第 1 列（如 max[1]）存放成績。圖 7-6 為搜尋運作程序，起先將最高成績的陣列設定為 0（max = {0, 0}）；最低成績設定成超過成績界線（min = {0, 999}），再依序比較每位學生成績，如果高於 max，則該紀錄複製到 max 陣列內；如果低於 min，則複製到 min 上。如此由第 0 筆到最後一筆搜尋之後，max 與 min 陣列分別得到最高與最低成績的紀錄。虛擬碼提示如下：

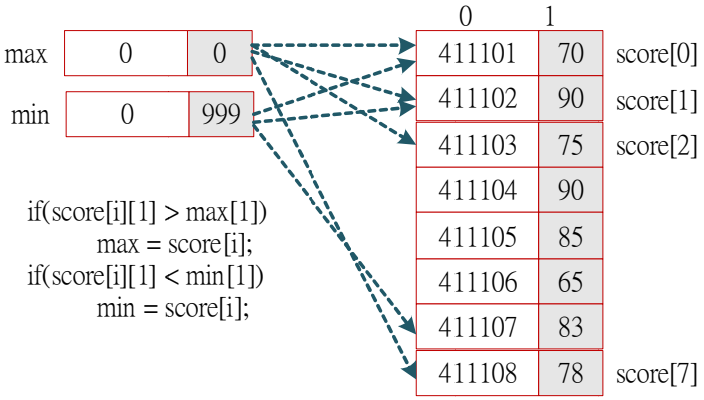

圖 7-11 最高與最低成績搜尋

```
01  宣告成績儲存陣列並給初值（int score[][]= {{411101, 70}, …} ）;
02
03  宣告最高成績陣列（int max[] = {0, 0} ）;
04  宣告最低成績陣列（int min[] = {0, 9999} ）;
05
06  搜尋最高與最低成績者：
07          for(int i=0; i<score.length; i++) {
08              if(score[i][1] > max[1])
```

```
09                          max = score[i];
10                  if(score[i][1] < min[1])
                            min = score[i];
                }
```

7-4-5 自我挑戰：成績查詢系統

（A）程式功能：PM7_7.java

數學老師利用一個二維陣列儲存某一班級學生的成績，score[][] = {{411101, 70}, {411102, 80}, {411103, 75}, {411104, 90}, {411105, 85}, {4111106, 65}, {411107, 83}, {411108, 78}}。請編寫程式可允許輸入學生學號，並輸出該學生的數學成績，期望操作介面如下：

```
D:\Java2_book\chap3>java PM3_2
== 成績查詢系統 ==
請輸入學生學號 =>4111120
沒有學號 = 4111120 的資料

D:\Java2_book\chap3>java PM3_2
== 成績查詢系統 ==
請輸入學生學號 =>411103
學號 = 411103, 成績= 75
```

（B）製作技巧提示：

利用輸入學號作為查詢關鍵，由陣列的起頭到結尾依序比較第 0 個欄位（score[i][0], i=0, 1, 2, …, score.length-1），如果相同則印出第 1 個欄位的內容（score[i][1]，成績欄位）；虛擬碼提示如下：

```
宣告輸入物件（Scanner）;

宣告成績儲存陣列並給初值（int score[][]= {{411101, 70}, …} ）;

宣告旗號並給予初值（int flag=0 ）;

讀入查詢學號（num ）;

搜尋查詢成績：
        for(int i=0; i<score.length; i++) {
            if(score[i][0] == num){
                輸出顯示學號與成績；
```

```
13                          flag = 1;
                        }
                    }
                    if(flag ==0)
                        輸出顯示沒有資料；
```

7-5 泡沫排序法

所謂排序法（Sort）即是將一大堆資料，利用某一關鍵內容由最大到最小，或最小到最大依序排列。吾人可能會認為排序演算法應該不是很重要才對，如果僅排序 100 筆以下資料，那用什麼演算法都差不了多少；但如果排序一千筆甚至幾十萬筆以上資料時，演算法的排序速度快或慢就變成很重要了。在計算機科學裡有許多排序的演算法，各種演算法都有其優缺點，本書僅介紹較常用的『泡沫排序法』，至於其他演算法，請參考有關資料結構的書本。

排序法那麼重要嗎？簡單的運用如學生成績排序、暢銷產品排列、營業員業績排行榜...等等，確實在生活領域裡隨時都會遇到排序的問題。但在計算機科學裡還有一個更重要的運用，即是如欲由幾千萬筆資料內查詢或變更某一筆資料，如何才能以最快速度搜尋到該筆資料，就牽涉到資料排列的問題，即是資料結構的構思。試想，如欲由一堆雜亂無章的資料內，找尋某一筆資料，若只能由到頭到尾一筆一筆的比對（順序搜尋法），運氣好，第一筆就找到；運氣差，最後一筆才找到，或發現根本沒有該資料。如果能將所有資料依照大到小或小到大排序，欲搜尋其中一筆資料也許會快許多；此運用方法，於本章 7-7 節中有範例說明。

7-5-1 泡沫排序演算法

『泡沫排序法』這種最普遍的排序演算法，是最不聰明卻最簡單的方法。圖 7-7 為其運作程序（由大到小排序），由陣列第 1 元素開始（a[0], i =0），作為基準並一個接一個比較所有其他元素，如果比 a[0] 大的話，則兩者交換。當所有資料比對完之後，最後結果是 a[0] 是所有元素之間的最大值；如此表示第 1 個泡沫已浮上來。接著，以第 2 個元素（a[1]）為基準，往下比較所有元素，如果較大的話，兩者相交換。往下比對完之後，最後 a[1] 內容會最大，如此表示第 2 個泡沫已浮上來。依此類推，總共需比較陣列元素 – 1 次輪迴（i - 1），最後即可得到由大到小的排序；另外，由小到大也是如此。

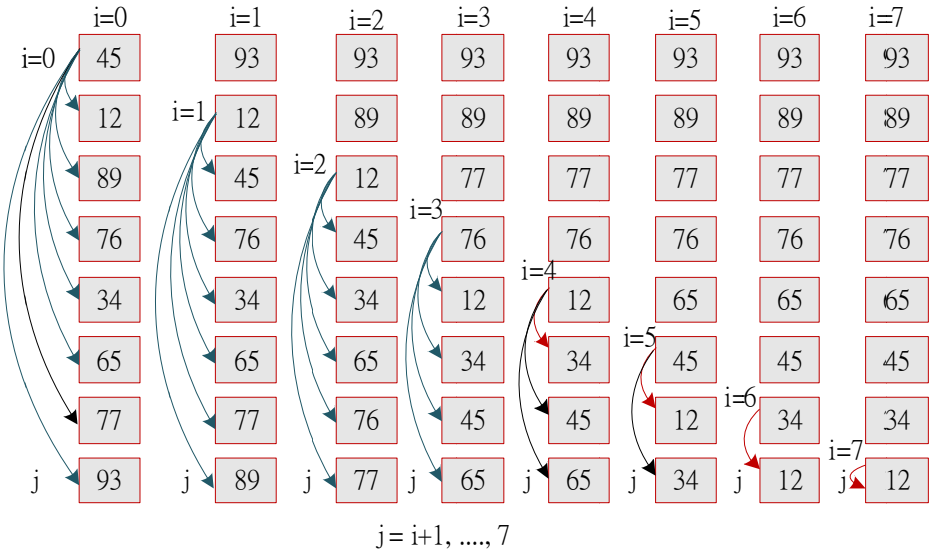

圖 7-12 泡沫排序法的運作程序

7-5-2 範例研討：成績高低排序

（A）程式功能：Ex7_8.java

張老師利用一維陣列存放了班上數學成績，陣列內容為 a [] = {45, 12, 89, 76, 34, 65, 77, 93, 70, 65, 45, 89}，請利用泡沫排序法編寫一程式由高到低排列，並印出每回合排列的結果；期望操作介面如下：

```
D:\Java1_book\chap7>java Ex7_8
陣列內容 : 45  12  89  76  34  65  77  93
第 0 回合: 93  12  45  76  34  65  77  89
第 1 回合: 93  89  12  45  34  65  76  77
第 2 回合: 93  89  77  12  34  45  65  76
第 3 回合: 93  89  77  76  12  34  45  65
第 4 回合: 93  89  77  76  65  12  34  45
第 5 回合: 93  89  77  76  65  45  12  34
第 6 回合: 93  89  77  76  65  45  34  12
第 7 回合: 93  89  77  76  65  45  34  12
```

（B）製作技巧研討：

吾人可利用雙重迴圈製作泡沫排序法的運作程式(如圖 7-7 所示)，外迴圈指定第幾回合(i=0, 1, .., a.length)，每回合找出一個較大的數值；內迴圈指定每回合所比較的元素(j = i+1, …, a.length)。

如果被比較元素大於指定元素 (a[j] > a[i])，則兩元素交換，否則不做任何處理。其中較困難的是，兩個元素 (或稱兩個變數) 的內容如何交換，這也是程式設計中較有趣的問題。兩變數內容交換的情況，就好像有兩個杯子，一個裝啤酒，另一杯裝可樂，這兩個杯子所裝的東西如何交換？其實非常簡單，我們只要再準備第三個杯子 (另一個變數)，先將第一個杯子的內容 (如啤酒) 倒入第三個杯子，再將第二杯 (如可樂) 倒入第一杯，最後再將第三杯 (如啤酒) 倒入第二杯，如此就完成第一杯與第二杯內容交換。需聲明一下，程式設計並非用倒入，而是用複製的，如圖 7-8 所示。

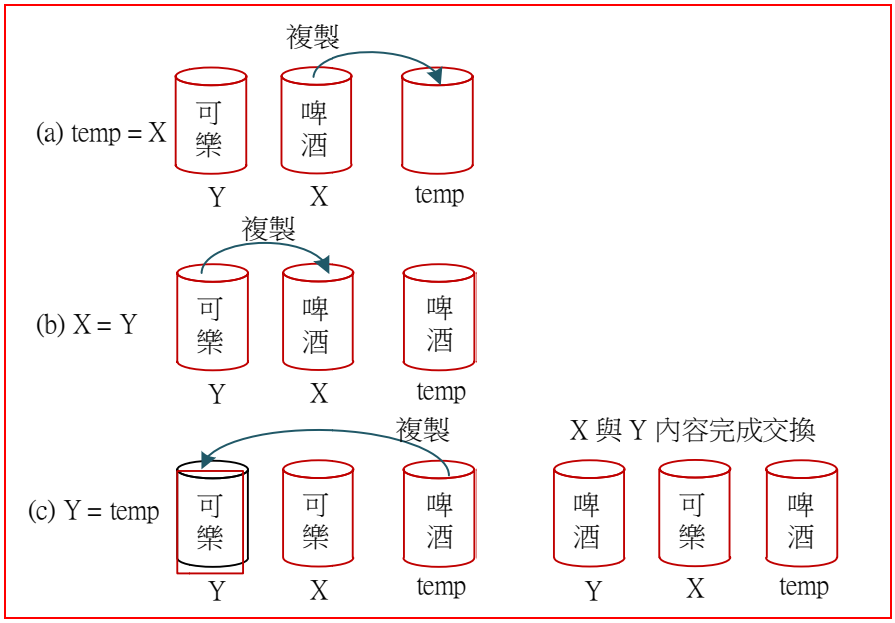

圖 7-13 兩變數內容交換的運作程序

(C) 程式範例：

```
01  //Ex7_4.java
02
03  public class Ex7_4 {
04        public static void main(String args[]) {
05              int a[] = {45, 12, 89, 76, 34, 65, 77, 93};
06              int temp;
07
08              System.out.printf("陣列內容 : ");    //顯示原來陣列內容
09              for(int j=0; j<a.length; j++)
10                    System.out.printf("%d    ", a[j]);
11              System.out.printf("\n");
12
13              for (int i=0; i<a.length; i++) {
14                    for (int j=i+1; j<a.length; j++) {
15                          if (a[i] < a[j]) {
16                                temp = a[i];
17                                a[i] = a[j];
```

```
18                          a[j] = temp;
19                      }
20                  }
21                  System.out.printf("第 %d 回合: ", i);
22                  for(int j=0; j<a.length; j++)
23                      System.out.printf("%d  ", a[j]);
24                  System.out.printf("\n");
25              }
26          }
27      }
```

（C） 程式重點說明：

(1) 第 13~25 行：『**for(int i=0; i<a.length; i++) { ….}**』。外迴圈，指定排序的回合。

(2) 第 14~20 行：『**for(int j=i+1; j<a.length; j++) {…}**』。內迴圈，指定每回合依序比較的元素。

(3) 第 15~19 行：『**if(a[i] <a[j] { …})**。基準元素比被比較元素小的話，則兩元素內容交換。

(4) 第 16~18 行：『**temp=a[i]; a[i] = a[j]; a[j]=temp**』。利用 temp 變數，將元素內容交換。

7-5-3 自我挑戰：列印股票高低排序

（A）程式功能：PM7_8.java

延續 PM3_1 範例，增加一項功能；其利用陣列 stock[] = {78, 72, 61, 56, 87, 66, 74, 88, 76, 58, 65, 57, 90, 78, 67, 89, 56, 77, 56, 87, 67, 80, 77, 86, 67, 75, 86, 98, 88, 78}; 儲存台積電最近 30 個交易日的收盤價，請依照價格高低順序（由最低到最高順序）印出其內容；期望操作介面如下：

```
G:\Examples\chap7>java PM7_9
原股票排序順序 :
78.0  72.0  61.0  56.0  87.0  66.0  74.0  88.0  76.0  58.0
65.0  57.0  90.0  78.0  67.0  89.0  56.0  77.0  56.0  87.0
67.0  80.0  77.0  86.0  67.0  75.0  86.0  98.0  88.0  78.0

排序後股票順序:
56.0  56.0  56.0  57.0  58.0  61.0  65.0  66.0  67.0  67.0
67.0  72.0  74.0  75.0  76.0  77.0  77.0  78.0  78.0  78.0
80.0  86.0  86.0  87.0  87.0  88.0  88.0  89.0  90.0  98.0
```

（B）製作技巧提示：

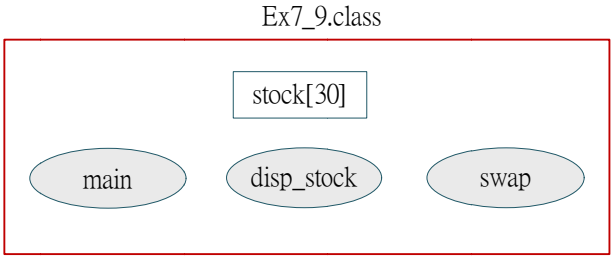

圖 7-14

原 PM7_1 範例將陣列 stock[] 宣告成類別變數，同一類別（PM7_6.class）內所有方法（如 main()、disp_stock()、swap()）都可以直接引用它。因此，本範例將兩元素交換的處理程序製作成 swap(i, j) 函數方法，功能是 stock[i] 與 stock[j] 兩元素內容互相交換。程式虛擬碼提示如下：

```
宣告類別陣列變數（static double stock[]={78, 72,…}）;
主方法範圍（main()）:
        呼叫列印陣列函數（disp_stock()）;
        泡沫排序演算法（需呼叫 swap(i, j) 函數）;
        呼叫列印陣列函數（disp_stock()）;
列印陣列函數範圍（static void disp_stock()）:
        for(int j=0; j<stock.length; j++) {
                System.out.printf("%.1f    ", stock[j]);
                if((j+1)%10 == 0)
                        System.out.printf("\n");
        }
元素交換函數範圍（static void swap(int x, int y)）:
        double temp;
        temp = stock[x];
        stock[x] = stock[y];
        stock[y] = temp;
```

7-5-4 自我挑戰：印出數學成績單

（A）程式功能：PM7_9.java

數學老師利用一個二維陣列儲存某一班級學生的成績，score[][] = {{411101, 70}, {411102, 80}, {411103, 75}, {411104, 90}, {411105, 85}, {4111106, 65}, {411107, 83}, {411108, 78}, {411109, 67}, (411110, 72)}，請依照成績由最高排列到最低印出。期望操作介面如下：

```
G:\Examples\chap7>java PM7_7
```

```
== 列印排序後成績單 ==
411104 = 90
411105 = 85
411107 = 83
411102 = 80
411108 = 78
411103 = 75
411101 = 70
411106 = 65
```

（B）製作技巧提示：

較特殊的地方是泡沫排序法中兩個元素交換程序，必須整筆資料交換，而不是僅成績交換；因此，暫存變數 temp 需宣告成每筆資料的型態（int[2]）。演算法重點如下：

```
int[] temp = new int[2];
for(int i=0; i<score.length; i++) {
        for(int j=i+1; j<score.length; j++) {
            if(score[i][1] < score[j][1]) {
                    temp = score[i];
                    score[i] = score[j];
                    score[j] = temp;
            }
        }
}
```

7-6 陣列資料結構

7-6-1 陣列資料結構的型態

資料大多是由多筆紀錄所構成，每筆資料再利用若干個欄位來描述其意思（第 8 章將詳細說明）。我們也可以利用二維陣列來儲存資料，陣列中每一行代表一筆紀錄，每行裡包含若干個欄位（列），存放某一特殊訊息，這就是陣列資料結構的架構。陣列資料結構最大的缺點是有元素都必須是相同的資料型態；當然還有其他表示法，以後再詳細介紹。我們用一般學生成績檔案為範例，來說明陣列資料結構的處理過程，如圖 7-9 所示。每一行儲存一筆學生資料，每行有 6 個欄位（6 列），分別存放學生學號、程式設計、資訊網路、資料處理、統計分析、以及總平均分數。

學 號	程式設計	資訊網路	資料處理	統計分析	總平均
492101	80	90	78	72	0
492102	90	67	54	46	0

492103	71	64	83	53	0
492104	68	89	73	73	0
492105	56	72	53	49	0
492106	96	65	78	83	0
492107	49	62	91	68	0

圖 7-9 陣列資料結構範例

7-6-2 範例研討：印出班級成績單

（A）程式功能：Ex7_9.java

張老師利用二維陣列 score[][] = {{492101, 80, 90, 78, 72, 0}, {492102, 90, 67, 54, 46, 0}, {492103, 71, 64, 83, 53, 0}, {492104, 68, 89, 73, 73, 0}, {492105, 56, 72, 53, 49, 0}, {492106, 96, 65, 78, 83, 0}, {492107, 49, 62, 91, 68, 0}} 存放該班的學生成績資料，每筆資料為 {學號、程式設計、資訊網路、資料處理、統計分析、總平均}。請計算每一位同學的總平均分數，再列印出所有成績資料。期望操作介面如下：

```
D:\Java1_book\chap7>java Ex7_9
學號  程式設計  資訊網路  資料處理  統計分析  總平均
492101   80       90       78       72       80
492102   90       67       54       46       64
492103   71       64       83       53       67
492104   68       89       73       73       75
492105   56       72       53       49       57
492106   96       65       78       83       80
492107   49       62       91       68       67
```

（B）製作技巧研討：

首先計算每位學生的總平均分數，亦填入該欄位（socre[k][5]，第 k 位學生），最後在印出所有內容即可。

（C）程式範例：

```
01 //Ex7_5.java
02
03 public class Ex7_5 {
04       public static void main(String args[]) {
05
06          int score[][] = {{492101, 80, 90, 78, 72, 0},
```

```
07                          {492102, 90, 67, 54, 46, 0},
08                          {492103, 71, 64, 83, 53, 0},
09                          {492104, 68, 89, 73, 73, 0},
10                          {492105, 56, 72, 53, 49, 0},
11                          {492106, 96, 65, 78, 83, 0},
12                          {492107, 49, 62, 91, 68, 0}};
13
14          /* 計算每位學生的總平均分數 */
15          for(int i=0; i<score.length; i++)
16              score[i][5] = (score[i][1]+score[i][2]+score[i][3]+score[i][4])/4;
17
18          /* 列印所有學生成績資料 */
19          System.out.printf("學號 程式設計 資訊網路 資料處理 統計分析 總平均\n");
20              for (int i=0; i<score.length; i++) {
21                  for(int j=0; j<score[i].length; j++)
22                      System.out.printf("%d \t", score[i][j]);
23                  System.out.printf("\n");
24              }
25      }
26  }
```

(D) 程式重點分析：

(1) 第 15~16 行：『for(int i=0; i<score.length; i++){ …}』。計算每筆資料的總平均分數。

(2) 第 20~24 行：『for (int i=0; i<score.length; i++) { …}』。列印出陣列所有內容。

7-6-3 自我挑戰：印出已排序成績單

（A） 程式功能：PM7_10.java

延續 Ex7_5 範例，請編寫一程式列印出依照總平均分數，由高至低順序的成績單，期望操作介面如下：

```
G:\Examples\chap7>java PM7_8
學號  程式設計  資訊網路  資料處理  統計分析  總平均
492101   80      90      78      72      80
492106   96      65      78      83      80
492104   68      89      73      73      75
492103   71      64      83      53      67
492107   49      62      91      68      67
492102   90      67      54      46      64
492105   56      72      53      49      57
```

（B）製作技巧提示：

以總平均為基準的排序是比較總平均分數，如果大於的話，則整筆資料都要交換；因此，暫存器需要宣告成具有 6 個欄位的一維陣列（int[] temp = new int[6]，每筆資料有 6 個欄位）。泡沫排序法提示如下：

```
/* 泡沫排序法, 依照總平均分數由高至低排列 */
int temp[] = new int[6];
for(int i=0; i<score.length; i++) {
        for(int j=i+1; j<score.length; j++) {
                if(score[i][5] < score[j][5]) {
                        temp = score[i];
                        score[i] = score[j];
                        score[j] = temp;
                }
        }
}
```

7-7 專題製作：陣列運用

7-7-1 範例研討：二分搜尋演算法

『二分搜尋法』是由已排序(由大到小或由小到大排列)的陣列裡，搜尋某一筆資料，圖 7-10 為其運作程序。假設 a 陣列已由小到大排序完成(a[low] ~ a[high])，key 變數儲存欲尋找的資料，首先比對 a 陣列的中間元素(mid)的內容，如果 key = a [mid]，則資料找到；如果 key < a[mid]，則表示資料位於 a[low] 與 a[mid] 之間，則 mid 取代 high；如果 key > a[mid]，則資料位於 a[mid] ~ a[high] 之間，mid 取代 low 內容。沒有找到資料的話，則將搜尋目標移到前半段或後半段繼續尋找，但還是由 a[low] ~ a[high] 搜尋(low 或 high 可能會改變)；如此依此類推，一直到找到資料或搜尋完畢為止。由此可見，此演算法每次將所有陣列資料缺切一半，再決定位於前段或後段，因此稱之為『二分搜尋法』。

此演算法較困難的地方是，如果找不到資料，如何去判斷與結束。其實，我們可以掌握一個重點，當 high 還是大於 low 的時候，表示還有資料未搜尋到；如果 high 小於或等於 low 時，表示已搜尋完所有資料，還未找到資料，則表示沒有該筆資料。

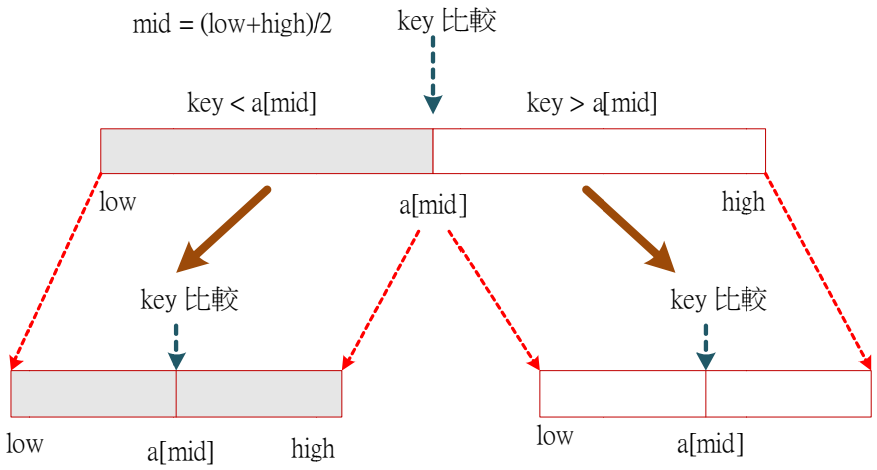

圖 7-15 二分搜尋法

（A）程式功能：Ex7_10.java

數學老師利用一個二維陣列 score[][] 儲存某一班級學生的成績，陣列第一個元素 score[k][0] 存放學生學號，由 411101 ~ 411150；第二個元素 score[k][1] 存放數學成績，由 00 ~ 100 分，如：{411101, 70}, {411102, 80}, {411103, 75}, {411104, 90}, {411105, 85}, {4111106, 65}, {411107, 83}, {411108, 78}, {411109, 67}, (411110, 72)...等等。陣列是依照學號由低到高排列。請編寫一程式，允許輸入學生學號，則輸出該學生的成績。程式中先寫一小程式填入陣列內每一位同學的學號與成績，成績採 00 ~ 100 之間的亂數，之後印出全班成績，接著再編寫一個二分搜尋程式。期望操作介面如下：

```
D:\Java1_book\chap7>java Ex7_10
=== 411101 ~ 411150 成績列印 ===
  90   67   81   51   97   73   56   59   19   57
  93   10   80   89   80   79   84   41   81   75
  89   28   86   71   75   34    3   74   79   83
  32   36   84   57   60   41   17   41   29   76
  88   70   64   94   99   49   36   58   58   56
請輸入欲尋找的學生學號 => 411102

學號 411102 成績是 67
```

（B）製作技巧研討：

本程式大略分為 3 個部分。第 1 部分填入學號與成績 (產生亂數)，第 2 部分是列印全班成績，第 3 部分，讀入尋找學號，再利用二分搜尋法找出該學號的成績。

（C）程式範例：

```
01 //Ex7_6.java
```

```
02
03 import java.util.Scanner;
04 import java.lang.Math;
05 public class Ex7_6 {
06         public static void main(String args[]) {
07                 Scanner keyin = new Scanner(System.in);
08                 int a[][], value, flag, low, high, mid;
09                 a = new int[50][2];
10                 int number = 411101;
11                 for(int i=0; i<a.length; i=i+1){
12                         a[i][0] = number + i;
13                         a[i][1] = (int)(Math.random()*100);
14                 }
15
16                 /* 列印全班成績 */
17
18                 System.out.printf("=== 411101 ~ 411150 成績列印 ===\n");
19                 for(int i=0; i<a.length; i++) {
20                         System.out.printf("%4d", a[i][1]);
21                         if((i+1) % 10 == 0)
22                                 System.out.printf("\n");
23                 }
24
25                 System.out.printf("請輸入欲尋找的學生學號 => ");
26                 value = keyin.nextInt();
27
28                 /*     二分搜尋法 */
29                 low = 0; high = 49;mid=0;
30                 flag=0;                                 // 設定是否找到旗號
31                 while((low+1) < high) {                 // 陣列是否搜尋完
32                         mid = (low + high)/2;
33                         if (value == a[mid][0]) {       // 比對陣列中間元素
34                                 flag = 1;               // 找到了, 設定旗號並離開
35                                 break;
36                         }
37                         else if (value > a[mid][0])     // 在陣列的後半段
38                                 low = mid;
39                         else                            // 在陣列的前半段
40                                 high = mid;
41                 }
42
43                 if (flag == 1)
44                    System.out.printf("學號 %d 成績是 %d\n", a[mid][0],
45
```

```
            a[mid][1]);
                    else
                        System.out.printf("沒有 %d 學號學生\n", value);
            }
        }
```

(D)程式重點分析：

(1) 第 28~41 行：**二分搜尋演算法**。

(2) 第 29 行：**『low=0; high=49;』**。設定原陣列搜尋範圍。

(3) 第 30 行：**『flag=0』**。設定搜尋旗號為否（還未找到）。

(4) 第 31~41 行：**『while((low+1) < high) { ….}』**。其中 (low+1) < high 表示元素較高的指標 high 還是高於較低的 low，則陣列還未搜尋完畢。

(5) 第 32 行：**『mid = (high + low)/2』**。陣列元素索引 high 與 low 之間的中間元素的索引 mid。

(6) 第 33 行**『if(value == a[mid] { …}』**。成立的話，表示找到該資料，則設定旗號並中斷離開。

(7) 第 37 行：**『else if (value > a[mid][0]) { …}』**。如未找到，但比中間值 a[mid] 大的話，則設定搜尋後半段（low = mid，但 high 未改變）。

(8) 第 39 行：**『else high = mid』**。都不是的話，則設定搜尋前半段（low 未改變）。

7-7-2 自我挑戰：大樂透對獎系統

（A）程式功能：PM7_11.java

許多客戶買了大樂透彩券，一直不瞭解開獎辦法，再說彩券購買太多的話，每次對獎都要花費許多時間。彩券公司為了方便客戶對獎，於是在網路上公布一套對獎系統，客戶可輸入當期開獎號碼，或觀察當期號碼，也可輸入所購買號碼，系統會告知是否得獎，或簽中哪一獎，大樂透開獎辦法是：利用氣球吹出 6 個 1~ 49 之間不重複的號碼，另外多一個特別號；中獎種類有：

中獎獎項	開獎辦法
頭講	六個號碼相同者。
貳獎	簽中五個號碼與特別號。
參獎	簽中五個號碼。

肆獎	簽中四個號碼與特別號。
伍獎	簽中四個號碼。
陸獎	簽中三個號碼與特別號。
普獎	簽中三個號碼。

期望操作介面格式如下：

```
G:\Examples\chap7>java PM7_9

==  歡迎光臨    大樂透對獎系統  ==
(1)  輸入開獎號碼  (2)  顯示開獎號碼
(3)  輸入對獎號碼  (4)  離開系統
                請選擇工作項目  =>1
請輸入  6  個開獎號碼  =>11 22 27 29 35 40
請輸入特別號碼  =>48

==  歡迎光臨    大樂透對獎系統  ==
(1)  輸入開獎號碼  (2)  顯示開獎號碼
(3)  輸入對獎號碼  (4)  離開系統
                請選擇工作項目  =>2
本期開獎號碼: 11 22 27 29 35 40
特別號: 48

==  歡迎光臨    大樂透對獎系統  ==
(1)  輸入開獎號碼  (2)  顯示開獎號碼
(3)  輸入對獎號碼  (4)  離開系統
                請選擇工作項目  =>3
請輸入  6  個對獎號碼  =>11 15 18 22 27 48
3  個對中號碼:11    22    27
特別號  = 48
恭喜您，陸獎

==  歡迎光臨  大樂透對獎系統  ==
(1)  輸入開獎號碼  (2)  顯示開獎號碼
```

```
(3) 輸入對獎號碼  (4) 離開系統
        請選擇工作項目 =>
```

（B）製作技巧提示：

吾人將開獎號碼與中獎號碼的陣列宣告成靜態變數，利用 4 個函數(方法)來製作整個系統。虛擬碼提示如下：

```
宣告開獎號碼、特別號變數（static int spaNum[], special; ）;
宣告中獎號碼、特別號與數目（static int winNum[], wining, winSpa; ）;
主方法範圍（main() ）:
    宣告對獎號碼陣列（int[] num = new int[6]; ）;
    顯示及讀入工作項目（select ）;
    While (select !=4) {
        switch(select) {
            case 1:
                讀入六個開獎號碼（spaNum[0], .. spaNum[5] ）;
                讀入特別號（special ）;
            case 2:
                顯示開獎號碼（spa_disp() ）;
            case 3:
                讀入六個對獎號碼（num[] ）
               /*  對獎程式  */
                wining = 0; winSpa=0; int k=0;
                for(int i=0; i<6; i++) {          // 6  個對獎號碼
                    for(int j=0; j<6; j++) {      // 6  個開獎號碼
                        if(num[i] == spaNum[j]) {
                            winNum[k] = num[i]; //  存入中獎陣列
                            wining = wining+1;    //  累增中獎數
                            k = k+1;
                        }
                    }
                    if(num[i] == special)          // 1  個特別號
                        winSpa = num[i];
                }
                /*  比對對獎數目
                switch(wining) {
                    case 0:
```

```
                    顯示全沒簽中；
                case 1:
                    呼叫顯示中獎號碼（win_disp()）
                    顯示沒有簽中；
                case 2:
                    呼叫顯示中獎號碼（win_disp()）
                    顯示沒有簽中；
                case 3:
                    呼叫顯示中獎號碼（win_disp()）
                    if (winSpa == 0)
                        顯示簽中普獎；
                    else {
                        顯示簽中陸獎；
                case 4:
                    呼叫顯示中獎號碼（win_disp()）
                    if (winSpa == 0)
                        顯示簽中伍獎；
                    else {
                        顯示簽中肆獎；
                case 5:
                    呼叫顯示中獎號碼（win_disp()）
                    if (winSpa == 0)
                        顯示簽中參獎；
                    else {
                        顯示中貳獎；
                case 6
                    呼叫顯示中獎號碼（win_disp()）;
                    顯示簽中頭講；
                }
            Default:
                顯示錯誤輸入，請重新選擇；
        }
        顯示及讀入工作項目（select）;
    }
宣告顯示工作項目方法（static void disp_menu()）;
宣告顯示開獎號碼及特別號方法（static void spa_disp()）;
宣告顯示中獎號碼及特別號方法（static void win_disp()）;
```