第二章　一維陣列

2-1 一維陣列

2-1-1 一維陣列結構

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 2-1 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(稱之有 8 個元素)，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)，並自動產生 a.length 表示該陣列元素的個數。

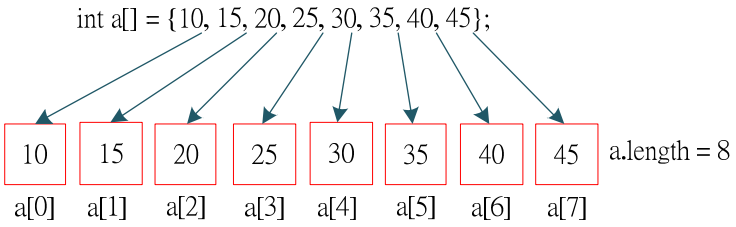

圖 2-1 一維陣列結構

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。

2-1-2 陣列動態宣告

『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 2-1 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：

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。

2-1-3 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。

2-2 一維陣列運用

2-2-1 範例研討：印出股票歷史價

（A）程式功能：Ex2_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：

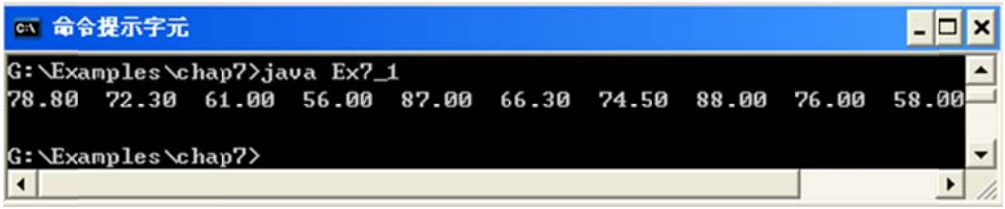

圖 2-1-1

（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  //Ex2_1.java
02
03  public class Ex2_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
12                  System.out.printf("\n");                  // 列印完畢，換行
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。

2-2-2 範例研討：印出平均股價

（A）程式功能：Ex2_2.java

請修改 Ex2_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   //Ex2_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 Ex2_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      }
    }
```

2-2-3 自我挑戰：印出最高與最低股價

（A）程式功能：PM2_1.java

請修改 Ex2_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 PM2_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 }
```

2-3 線性搜尋法

2-3-1 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(Linear Serach)；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。

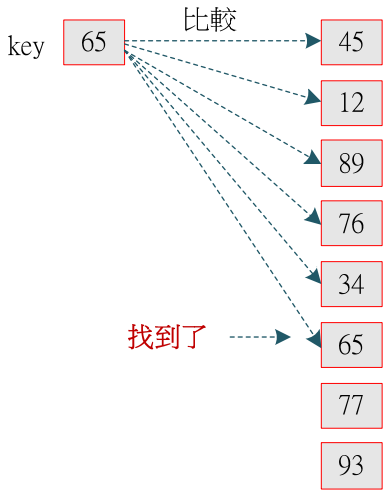

圖 2-2 線性搜尋法的運作

2-3-2 範例研討：實現線性搜尋法

（A） 程式功能：Ex2_3.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 Ex2_3 {
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]);
10              System.out.printf("\n");
11              System.out.printf("請輸入一個數值 =>");
12              value = keyin.nextInt();
```

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

2-3-3 範例研討：大樂透電腦選號

（A）程式功能：Ex2-4.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：

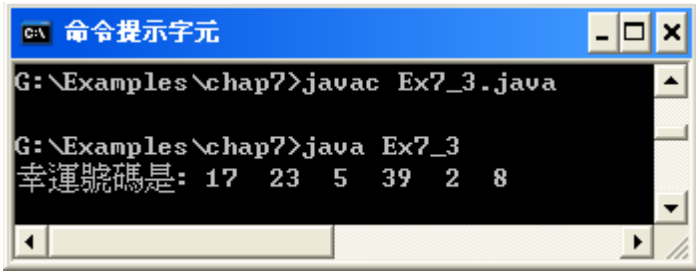

圖 2-2-1

（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。

（C）程式範例：

```
01  //Ex2-4.java
02
03  import java.lang.Math;
04  public class Ex2_4 {
05        public static void main(String args[]) {
06              int value;                          // 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

(1) 第 9~16 行：『**while{i<6} {…}**』。選出 6 個不重複號碼。

(2) 第 11~17 行：『**for(int j=0; j<i; j++) { if(value == num[j]) …}**』。線性搜尋敘述句。

(3) 第 13~14 行：『**if(value == num[j]) continue;**』。如發現號碼重複，則放棄重來。

2-4 泡沫排序法

2-4-1 泡沫排序演算法

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。

『泡沫排序法』是最普遍的排序演算法，也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。

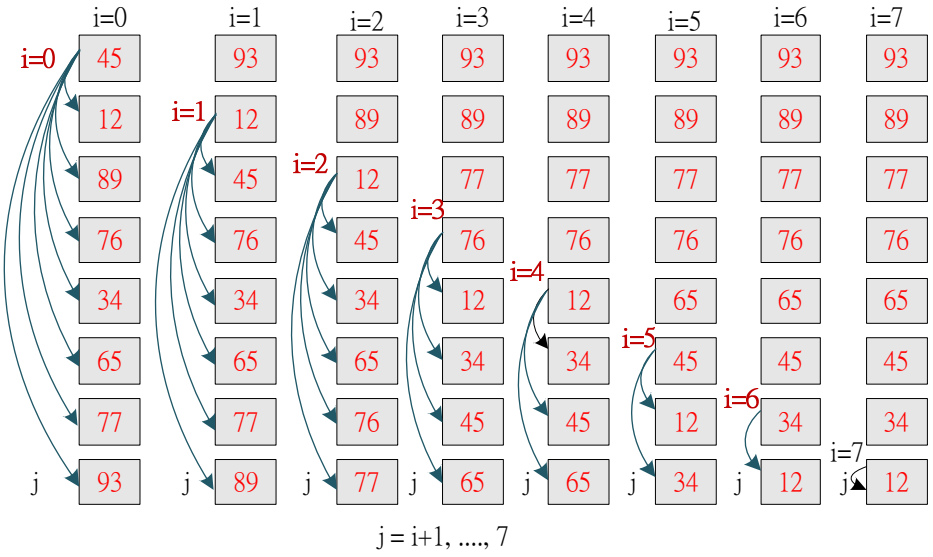

圖 2-3 泡沫排序法的運作程序

2-4-2 範例研討：成績高低排序

（A）程式功能：Ex2_5.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：

圖 2-3-1

（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 2-3 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。

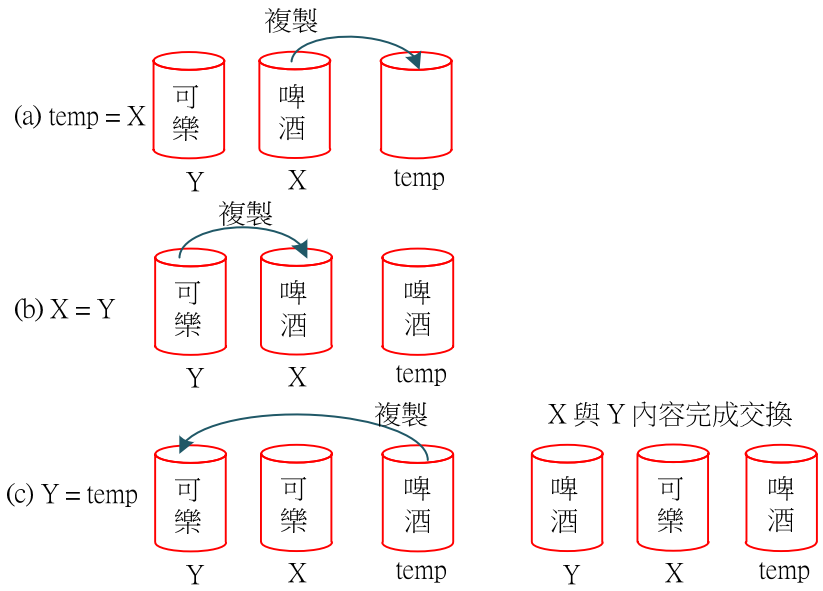

圖 2-4 兩變數內容交換的運作程序

（C）程式範例：

```
01  //Ex2_5.java
02
03  public class Ex2_5 {
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  }
```

(D) 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。

2-4-3 自我挑戰：列印股票高低排序

（A）程式功能：PM2_2

吾人利用陣列 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\chap2>java PM2_2
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：

此題目只要用泡沫排序法(如 Ex2_5.java)即可達成，但我們希望製作一只較通用型的系統(可擴充出 Ex2_6.java)，將兩元素交換的動作製作成一個 swap() 方法，並將陣列 stock[] 宣告成類別變數。圖 2-# 為此程式架構，其中 main()、disp_stock()、swap()) 都可以直接引用 stack[] 陣列。

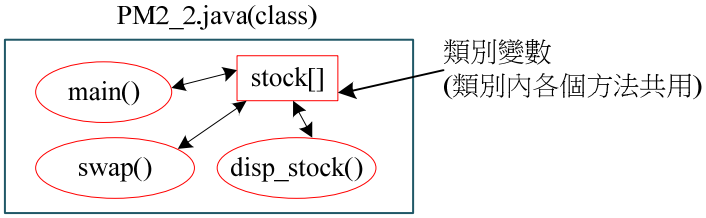

圖 2-5 PM2-2 程式架構

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()）;
        泊沫排序演算法（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;
}
```

2-5 專題研討

2-5-1 範例研討：記錄最近 30 天收盤價

（A）程式功能：Ex2_6.java

請設計一套程式，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 2-6），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) 程式設計技巧

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]。

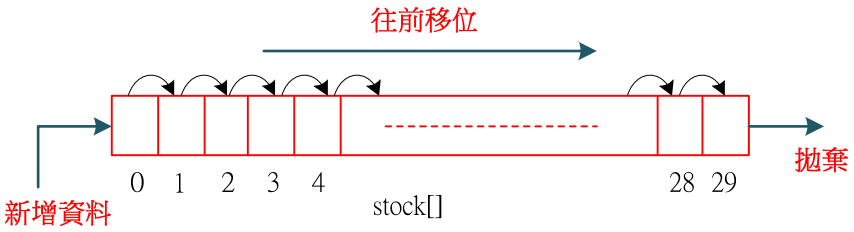

圖 2-6 紀錄過去 30 日收盤價

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。

(C) 程式範例

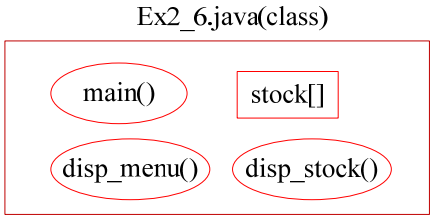

圖 2-7 Ex2_6 程式架構

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
47
48
49
50
```

```
                System.out.printf("\n");       //列印五筆, 換行
            }
            System.out.printf("\n");                  // 列印完畢, 換行
        }
    }
```

2-5-2 自我挑戰：股票走勢分析系統

（A）程式功能：PM2_3.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：

(a) 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 =>
```

(b) 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
```

(C) 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
```

(D) 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
```

(E) 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
```

(F) 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。

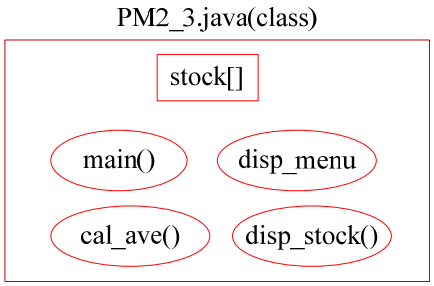

圖 2-7-1

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；
```

```
23                                  if ((cost>ave_15) && (cost>ave_30))
24                                          顯示多頭走勢；
25                                  else if    ((cost <ave_15) && (cost <ave_30)
26                                          顯示空頭走勢；
27                                  else if ((cost >ave_15) && (cost <ave_30))
28                                          顯示盤整轉向多頭走勢；
29                                  if ((cost>ave_15) && (cost>ave_30))
30                                          顯示盤整轉向空頭走勢；
31                      } // switch/case  結束
32                      顯示功能選項（disp_menu()）;
33                      讀取功能選項（select = keyin.nextInt()）;
34                  } // while  結束
35              } //  主方法結束
36              顯示選單方法範圍（static void disp_menu()）{.....}
37              計算  k  日平均價範圍（static double cal_ave(int k)){
38                  double sum=0;
39                  for(int I=0; I<k; I++)
40                      sum = sum + stock[I];
41                  return sum/k;
42              } // cal_ave()  結束
43               列印  30  日內所有股價範圍（static void disp_stock()）{
44                  for(int i=0; i<stock.length; i++) {
45                      System.out.printf("%.2f    ", stock[i]);
46                       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
```

2-5-3 自我挑戰：印製國字收據

(A)程式功能：PM2_4.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：

```
G:\Examples\chap4>java PM2_4
```

```
請輸入大德先生/小姐姓名 = >tsnien
請輸入捐款金額 =>345678
列印收據如下:

****     真自在遊民收容所     捐款收據 ****

        感謝 tsnien 先生/小姐大德贊助
                    捐款 345678 元整

總計 = 零 仟 零 佰 參 拾 肆 萬 伍 仟 陸 佰 柒 拾 捌 元整

**** 四海之內皆兄弟 順祝 平安快樂 ****
```

（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。

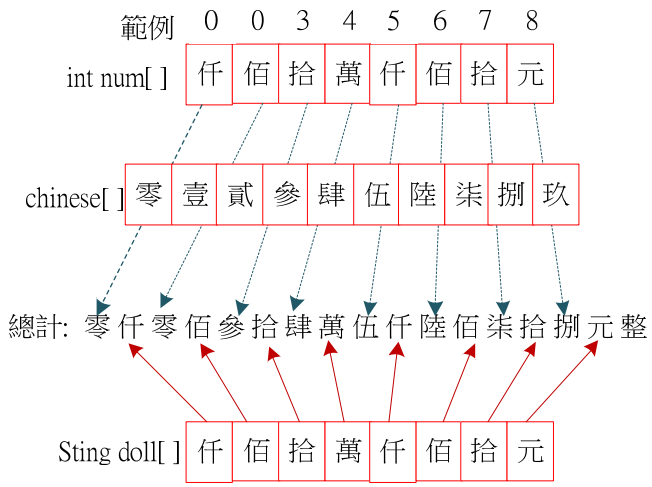

圖 2-8 國字列印格式

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：

```
宣告國字數字陣列（String chinese[] = {"零", "壹", "貳", "參", .., "玖"}）;
宣告列印格式陣列（String doll[] = {"仟", "佰", "拾","萬", “仟”, “佰”, “拾”, "元"}；
宣告位數數字陣列（int[] num = new int[9]）;
宣告最高數字變數（千萬，base=10000000）;
讀取捐款姓名（name）;
讀取捐款額（value）;
備份捐款額（value1 = value）;
取出各個位數的數字：(最高 8 位數)
        for (int i=0; i<8; i++) {        //i=0, 仟萬; i=1, 佰萬; i=2, 拾萬; …
            num[i] = value / base;
            value = value % base;
            base = base /10;
        }
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");
```

2-5-4 自我挑戰：統計滿意度平均值

(A) 程式功能：PM2_5.java

歡樂歌唱比賽邀請了 10 位老師當評審員，每位老師可給予 0~10 分，但記分方法是捨棄最高與最低分數後，再計算其餘 8 位老師的平均分數。期望操作介面，如下：

```
D:\Java2_book\chap2>javac PM2_5.java

D:\Java2_book\chap2>java PM2_5
請輸入第 1 位評審分數(0 ~ 10)=>5
請輸入第 2 位評審分數(0 ~ 10)=>1
請輸入第 3 位評審分數(0 ~ 10)=>6
請輸入第 4 位評審分數(0 ~ 10)=>7
```

```
請輸入第  5  位評審分數(0 ~ 10)=>9
請輸入第  6  位評審分數(0 ~ 10)=>9
請輸入第  7  位評審分數(0 ~ 10)=>8
請輸入第  8  位評審分數(0 ~ 10)=>7
請輸入第  9  位評審分數(0 ~ 10)=>6
請輸入第  10  位評審分數(0 ~ 10)=>7
有效評分: 5     6     6     7     7     7     8     9
平均分數為: 6.88
```

(B) 程式製作技巧

首先產生 10 個元素陣列 a[0]~a[9]，分別讀入每位老師的評分後，再利用泡沫排序法，將 a[] 陣列由小到大排序，並捨棄 a[0] 與 a[9]後，再計算分均分數。

(C) 程式提示

```
01  //PM2_5.java
02  ….
03  import java.util.Scanner;
04  public class PM2_5{
05        public static void main(String[] args){
06              Scanner in = new Scanner(System.in);
07              int[] a = new int[10];
08                 …..
09              while (count < 10) {
10                    System.out.printf("請輸入第  %d  位評審分數(0 ~ 10)=>", count+1);
11                    a[count] = in.nextInt();
12                    count++;
13              }
14            ……..
15      }
16  }
```