第六章
陣列

6.1 認識陣列與一般資料型態的不同
6.2 認識一維與二維陣列
6.3 學習陣列的應用

1

補充

陣列的用意

- 少量資料
  - 可以為每一筆資料設定一個變數
- 龐大資料
  - 譬如一個班級的50位同學成績
  - 分別設定A1、A2・・・A50等50個變數
  - 是一件相當麻煩的事。
- 為了解決大批的資料處理，遂有陣列(Array)的使用。

2

補充

陣列的用意

- 要儲存一個班級的數學成績可以宣告一個一維陣列，
  - 例如int A[50]代表50個人的數學成績
  - 則A[0]可以用來表示1號同學的數學成績
    (註：陣列索引由0開始)
  - 若要表達每班每人的國、英、數三科成績時，可以宣告一個二維陣int A[50][3]來表示
    - 其中A[0][0]表示1號國文成績
    - A[0][1]表示1號英文成績
    - A[0][2]表示1號數學成績
    - 依此類推。
    - 可以表示到A[49][2]為止。
  - 如果一年級有12班，則可以宣告個三維陣列int A[12][50][3]來儲存
    - 其中A[6][41][1]代表7班42號英文成績，以此類推。
  - 如果一個學校有三個年級則可宣告一個四維陣列int A[3][12][50][3]來儲存
    - 其中A[1][7][23][2]表示二年級八班24號數學成績。

3

6.1 一維陣列

一維陣列

- 一維陣列（1-dimensional array）可以存放多個相同資料型態的資料。
- 使用陣列必須經過兩個步驟：
  (1) 宣告陣列
  (2) 配置記憶體給該陣列
- 一維陣列的宣告與配置記憶體格式：

一維陣列的宣告與配置記憶體

```
資料型態 陣列名稱[];          // 宣告一維陣列
陣列名稱 = new資料型態[個數];// 配置記憶體給陣列
```

4

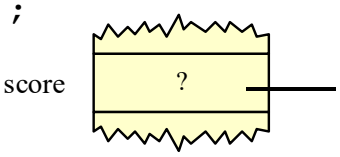
一維陣列的宣告及使用 (1/4)

- 下面的範例是一維陣列的宣告及記憶體配置：

```
01  int score[];          // 宣告整數陣列score
02  score=new int[4];  // 配置可存放4個整數的記憶體空間
```

- 執行完第1行後，編譯器會配置一塊記憶體給它，用來儲存指向陣列實體的位址：

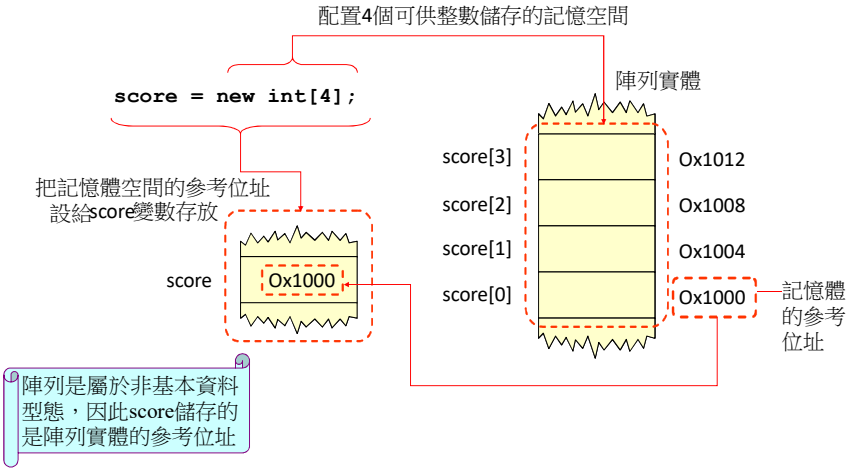
```
int score[];
```



5

一維陣列的宣告及使用 (2/4)

- 第2行是記憶體配置的動作：



6

一維陣列的宣告及使用 (3/4)

- 宣告一維陣列的另一種寫法：

宣告陣列的同時便配置記憶體

資料型態 陣列名稱[]=**new**資料型態[個數];

一維陣列的宣告及使用 (4/4)

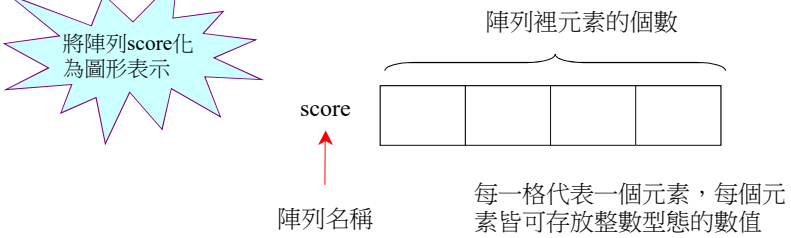
- 一維陣列的宣告範例：

```
int score[]=new int[4];
```

宣告一個整數陣列score，同時配置一塊可存放4個整數的連續記憶體空間

```
int score[]=new int[4];
```

將陣列score化為圖形表示



陣列的另一種宣告方式

- 稍早是以下列語法宣告score陣列：

```
int score[];              // 宣告score陣列為整數型態
```

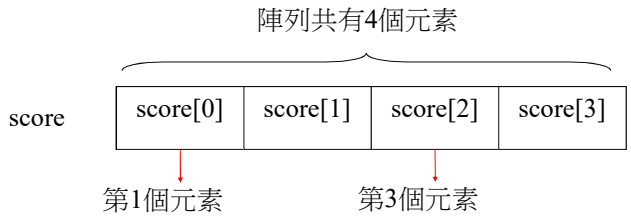
- 還可以用另一種語法宣告：

```
int[] score;              // 宣告score變數，其型態為整數陣列
```


陣列元素的表示方法

- 要存取陣列裡的元素，可以利用索引值（index）
- 陣列索引值的編號是由0開始
- 下圖為score陣列中元素的表示法及排列方式：

```
int score[]=new int[4];
```



動動腦1

- Q: int mtrx[ ]=new int[6];
  - 陣列名稱？
  - 可儲存的變數型態？
  - 陣列大小？

11

動動腦2

- 指出下列陣列宣告時，語法錯誤所在。
  - float height(35);
  - int for[]=new float[8];
  - int mtrx[12];

12

一維陣列的使用範例

- 下面的程式是一維陣列的使用範例：

```
01 // app6_1,一維陣列
02 public class app6_1
03 {
04   public static void main(String args[])
05   {
06     int i;
07     int a[];                  // 宣告整數陣列a
08     a=new int[3];             // 配置可存放3個整數的記憶體空間供整數陣列a使用
09     a[0]=5;                   // 設定第一個元素的值為5
10     a[1]=8;                   // 設定第二個元素的值為8
11
12     for(i=0; i<a.length; i++)          // 印出陣列的內容
13       System.out.print("a["+i+"]="+a[i]+",\t");
14
15     System.out.println("\nLength of array = "+a.length); // 印出陣列長度
16   }
17 }
```

```
/* app6_1 OUTPUT------
a[0]=5, a[1]=8, a[2]=0,
Length of array = 3
----------------------*/
```


下表為常用的跳脫字元：

表3.2.3 常用的跳脫字元

跳脫字元	所代表的意義	跳脫字元	所代表的意義
\f	換頁(Form feed)	\\	反斜線(Backslash)
\b	倒退一格(Backspace)	\'	單引號(Single quote)
\n	換行(New line)	\"	雙引號(Double quote)
\r	歸位(Carriage return)	\uxxxx	十六進位的unicode字元
\t	跳格(Tab)	\ddd	八進位Unicode字元，範圍在八進位的000~377之間

print() 列印之後不換行
println() 列印之後換行

陣列的長度

- 取得陣列元素的個數（陣列長度）的格式：

陣列長度的取得

```
陣列名稱.length
```

- 如下面的程式片段：

```
a.length          // 印出陣列的長度
```


陣列初值的設定 (1/2)

- 在宣告時就給與陣列初值的格式：

陣列初值的設定

```
資料型態 陣列名稱[]={初值1，初值2，…，初值n};
```

- 以上面的格式宣告時，會視初值的個數來決定陣列的長度，如下面的範例：

```
int day[]={31,28,31,30,31,30,31,31,30,31,30,31}; // 宣告並設定初值
```

陣列元素有12個，day[0]為31，day[1]為28，...，day[11]為31

動動腦3

- boolean boo []=new boolean[6];
- 請問在boo陣列裡的每一個元素，其預設值為多少？

17

3.2.5 基本資料型態的預設值

表3.2.5 基本資料型態的預設值

資料型態	預設值
byte	(byte)0
short	(short)0
int	0
long	0L
float	0.0f
double	0.0d
char	\u0000
boolean	false

18

動動腦4

- 請問下列初值設定方式正確嗎？
  - (1) int arr={6, 7, 8, 9};
  - (2) int arr[5]={5,1,1,1,1,1};
  - (3) int arr[]={1,1,1,1,1};

19

陣列初值的設定 (2/2)

- app6_2是一維陣列設定初值的範例：

```
01 // app6_2,一維陣列的設值
02 public class app6_2
03 {
04   public static void main(String args[])
05   {
06     int sum=0;
07     int a[]={15,6,8,7,12,7};                    // 宣告整數陣列a,並設定初值
08
09     for(int i=0;i<a.length;i++)          // 計算陣列元素的和
10       sum+=a[i];
11
12     System.out.println("Average "+(float)sum/a.length);
13   }
14 }
```

```
/* app6_2 OUTPUT--
Average 9.166667
----------*/
```

sum=sum+a[i]

注意：a.length的長度為6
所以符合條件的i包含：
a[0],a[1],a[2],a[3],a[4],a[5]，
共6個陣列元素

20

簡單的範例

- 下面的程式可以找出陣列裡元素的最大值及最小值：

```
01 // app6_3,比較陣列元素值的大小
02 public class app6_3
03 {
04   public static void main(String args[])
05   {
06     int i,min,max;
07     int a[]={74,48,30,17,62};        // 宣告整數陣列a,並設定初值
08
09     min=max=a[0];
10     System.out.print("Elements in array a are ");
11     for(i=0;i<a.length;i++)
12     {
13           System.out.print(a[i]+"  ");
14           if(a[i]>max)                // 判斷最大值
15              max=a[i];
16           if(a[i]<min)                // 判斷最小值
17              min=a[i];
18     }
19     System.out.println("\nMaximum is "+max);          // 印出最大值
20     System.out.println("Minimum is "+min);    // 印出最小值
21   }
22 }
```

```
/* app6_3 OUTPUT----------
Elements in array a are 74  48  30  17  62
Maximum is 74
Minimum is 17
----------*/
```


6.1 課堂練習(1/2)

- 試撰寫一個程式，計算一維陣列中最大與最小的差值。
- 檔名：**ex6_1_1.java**

```
/* output----------
陣列內容:12 54 39 88 5 51
最大值:88
最小值:5
88-5=83
----------*/
```

6.1 課堂練習(2/2)

- int array[]={3,5,0,3,2,4,1,6,8,5,4,3,2}
  - 以length計算出陣列個數
  - 找出陣列中值為3-6的元素
  - 計算3-6的元素個數
  - **檔名：ex6_1_2.java**

```
共有13個陣列個數
元素值介於3-6的陣列分別為：3  5  3  4  6  5  4  3
共有8個元素介於3-6
```

23

二維陣列的宣告

- 二維陣列的宣告與配置記憶空間的格式：

二維陣列的宣告格式

```
資料型態  陣列名稱[][];
陣列名稱=new  資料型態[列的個數][行的個數];
```

- 如下面的範例：

```
int score[][];
score=new int[4][3];
```

24

另一種宣告方式

- 以較為簡潔的方式來宣告陣列：

二維陣列的宣告格式

```
資料型態 陣列名稱[][]=new 資料型態[列的個數][行的個數];
```

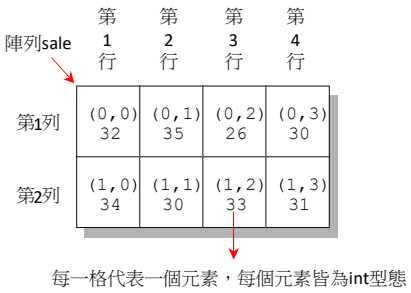
- 下面是二維陣列的宣告範例：

```
int score[][]=new int[4][3]; //宣告的同時即配置一塊記憶體空間
```


二維陣列的實例 (1/2)

- 下表為某汽車銷售公司的車輛銷售量：

業務員	2005年銷售量			
	第一季	第二季	第三季	第四季
1	32	35	26	30
2	34	30	33	31



- 上面的資料可用二維陣列儲存，宣告方式為

```
int sale[2][4];    // 宣告一個2列4行的整數陣列sale
```

二維陣列的實例 (2/2)

- 二維陣列的宣告與配置記憶空間的格式：

二維陣列初值的設定格式

```
資料型態 陣列名稱[][]={{ 第1列初值 },
                       { 第2列初值 },
                       {  ...      },
                       { 第n列初值 }};
```

```
int sale[][]={{30,35,26,32},// 二維陣列的初值設定
              {33,34,30,29}};
```

2×4 的陣列是由 2 個具有 4 個元素的一維陣列所組成

```
int sale[2][4]={{32,35,26,30},{34,30,33,31}};
```

2×4 的陣列

一維陣列，有 4 個元素

一維陣列，有 4 個元素

每列的元素個數不同的二維陣列

- matx[]為每列元素個數不同的二維陣列：

```
int matx[][]={ {31,12,14,11},
               {33,34,30},
               {12,81,32,14,17} };
```

- 宣告每列元素個數不同的二維陣列，但不設定初值：

```
int matx[][]=new int[3][];    // 宣告二維陣列，並指定列數
matx[0] = new int[4];         // 指定第一列有4個元素
matx[1] = new int[3];         // 指定第二列有3個元素
matx[2] = new int[5];         // 指定第三列有5個元素
```

28

取得列數與特定列之元素的個數

- 取得二維陣列的列數與特定列之元素的個數語法

取得二維陣列的列數與特定列之元素的個數

```
陣列名稱.length                  // 取得陣列的列數
陣列名稱[列的索引值].length      // 取得特定列元素的個數
```

- 如下面的程式片段：

```
matx.length              // 取得陣列 matx 的列數，其值為 3
matx[0].length           // 取得陣列 matx 的第 1 列元素的個數，其值為 4
matx[2].length           // 取得陣列 matx 的第 3 列元素的個數，其值為 5
```


二維陣列元素的引用及存取

- 下面是二維陣列的完整使用範例：

```
01  // app6_4,二維陣列的輸入輸出
02  public class app6_4
03  {
04    public static void main(String args[])
05    {
06      int i,j,sum=0;
07      int sale[][]={{32,35,26,30},{34,30,33,31}};//宣告陣列並設定初值
08
09      for(i=0;i<sale.length;i++)     // 輸出銷售量並計算總銷售量
10      {
11        System.out.print("業務員"+(i+1)+"的業績分別為 ");
12            for(j=0;j<sale[i].length;j++)
13            {
14            System.out.print(sale[i][j]+" ");
15            sum+=sale[i][j];
16            }
17            System.out.println();    // 列印換行
18      }
19      System.out.println("\n總銷售量為"+sum+"部車");
20    }
21  }
```

```
32 35 26 30
34 30 33 31
```

```
/* app6_4 OUTPUT----------
業務員1的業績分別為 32 35 26 30
業務員2的業績分別為 34 30 33 31

總銷售量為251部車
----------*/
```

6.2 課堂練習

- **ex6_2_1.java**
  - (1) 訂二維陣列如下，並列印出此二維陣列。
  - (2) 計算出此二維陣列中最大的值
  - (3) 表示出此最大值的索引值

```
[0][0]=3   [0][1]=5   [0][2]=0
[1][0]=3   [1][1]=2   [1][2]=4
5為最大的值，索引值為[0][1]
```

31

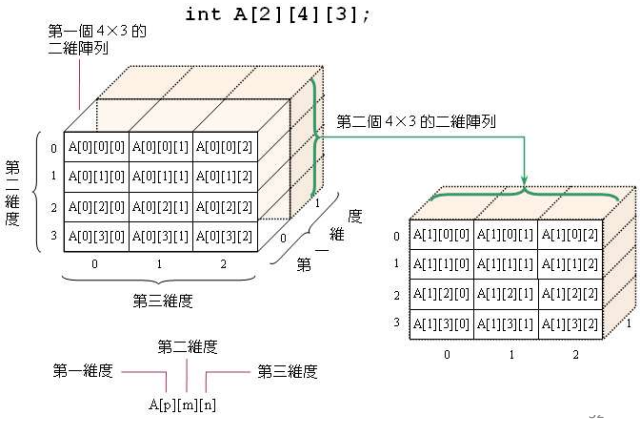
三維陣列

- 三維陣列的宣告範例：

```
int A[2][4][3];     // 宣告 2×4×3 整數陣列 A
```

2x4x3的三維陣列可看成是由2個4x3的二維陣列所組成

也就是兩組4個橫列，3個直行的積木併在一起，組成一個立方體



找出最大值的範例 (1/3)

下面的範例說明如何在三維陣列裡，找出所有元素的最大值：

```
01 // app6_5,
02 public class app6_5
03 {
04   public static void main(String args[])
05   {
06     int A[][][]={{{21,32,65},
07                   {78,94,76},
08                   {79,44,65},
09                   {89,54,73}},
10                  {{32,56,89},
11                   {43,23,32},
12                   {32,56,78},
13                   {94,78,45}}};
14
15     int i,j,k,max=A[0][0][0];          // 設定max為A陣列的第一個元素
16
17     for(i=0;i<A.length;i++)            // 外層迴圈
18       for(j=0;j<A[i].length;j++)       // 中層迴圈
19         for(k=0;k<A[i][j].length;k++)// 內層迴圈
20           if(max<A[i][j][k])
21             max=A[i][j][k];
22
23     System.out.println("max="+max);            // 印出陣列的最大值
24   }
25 }
```

設定2x4x3陣列的初值

利用三個for迴圈找出陣列的最大值

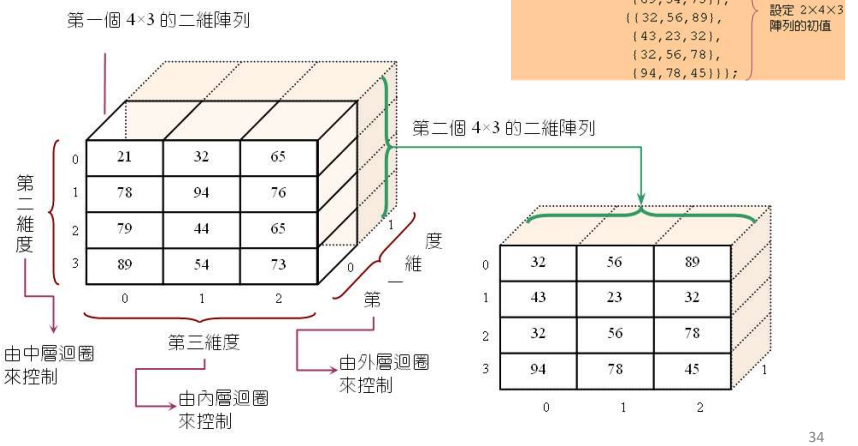
```
/* app6_5 OUTPUT-
max=94
-----------------*/
```


找出最大值的範例 (2/3)

- app6_5中三維陣列A的示意圖：

```
int A[][][]={{{21,32,65},
              {78,94,76},
              {79,44,65},
              {89,54,73}},
             {{32,56,89},
              {43,23,32},
              {32,56,78},
              {94,78,45}}};
```

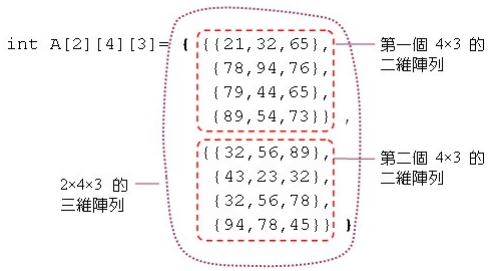
設定 2×4×3 陣列的初值



找出最大值 (3/3)

- 2x4x3的三維陣列可以寫成
  2x4x3的三維陣列 = { 4x3的二維陣列，4x3的二維陣列 }

- 陣列A初值的設定便可用下圖來表示：




```
package ex_9700837;
import java.util.*;
class h{
   public static void main(String[]args){
   Scanner input=new Scanner(System.in);
   int[] tall= {1,2,3,4,5};
   for (int i=0; i<tall.length; i++){
      System.out.print(tall[i]+" ");
   }
   System.out.println();
   System.out.println();
   System.out.println("另一種列印方法");

   for(int i : tall){
      System.out.print(i+" ");
   }
   }
}
```

```
1 2 3 4 5

另一種列印方法
1 2 3 4 5
```

6.3 課堂練習

- (1)列印出陣列
- (2) 找出下列陣列之最小值
- **ex6_3_1.java**

```
int A[][][]={{{15,85,36},{30,14,37},
              {47,23,96},{19,39,51}},
             {{22,16,51},{97,30,12},
              {68,77,26},{57,32,76}}};
```

```
/* output---
[0][0][0]=15 [0][0][1]=85 [0][0][2]=36
[0][1][0]=30 [0][1][1]=14 [0][1][2]=37
[0][2][0]=47 [0][2][1]=23 [0][2][2]=96
[0][3][0]=19 [0][3][1]=39 [0][3][2]=51
[1][0][0]=22 [1][0][1]=16 [1][0][2]=51
[1][1][0]=97 [1][1][1]=30 [1][1][2]=12
[1][2][0]=68 [1][2][1]=77 [1][2][2]=26
[1][3][0]=57 [1][3][1]=32 [1][3][2]=76

12為最小值
min=12
----------*/
```

37

回家作業

- hw6_1.java [陣列計算]
  1. 請撰寫一程式，由鍵盤輸入 10 個整數，並存放到一陣列。
  2. 程式執行時，顯示如下頁 的參考畫面，顯示【請 輸入10個整數：】，並顯示 【第1個整數：】，要求輸入第1個整數 。
  3. 依序要求輸入第1個至第10個整數，顯示執行結果如下頁。
  4. 判斷輸入10個整數後，計算陣列中大於 60有幾個，這些大於60的數值總合及平均值，顯示如下。

```
請輸入10個整數：
第1個整數：88
第2個整數：59
第3個整數：66
第4個整數：46
第5個整數：92
第6個整數：74
第7個整數：52
第8個整數：58
第9個整數：69
第10個整數：81
陣列中大於60的有6個
總合為470
平均值為78.33333333333333
```

38

回家作業

- hw6_2.java [浮點數計算]

1. 請撰寫一程式，由鍵盤輸入學生的人數，根據所輸入的學生人數，動態產生一個符合大小的浮點數陣列。
2. 將所輸入的每位學生成績存放到陣列裡(不限制輸入的小數點位數)。
3. 程式執行時，顯示【請輸入學生人數：】，要求輸入學生人數。
4. 接續要求輸入第1個至第n個學生的成績，n是剛才所輸入的學生人數。
5. 計算出人數、總分及平均值(不限制小數點位數)，顯示執行結果如下。

```
請輸入學生人數：5
第1個學生的成績：81.24
第2個學生的成績：56.14
第3個學生的成績：92.84
第4個學生的成績：42.96
第5個學生的成績：64.37
人數：5
總分：337.55
平均：67.509995
```

39

回家作業

- hw6_3.java [矩陣相加]

1. 試撰寫一個函數，此函數可用來計算矩陣 A 與 B 的和，使「矩陣之和」程式正常執行。

   int A[][] = {{1, 2, 3}, {4, 5, 6}};

   int B[][] = {{7, 8, 9}, {10, 11, 12}};
2. 計算矩陣 A 與 B 的和，並把相加後的結果放在矩陣 C 裡。
3. 顯示執行結果如下

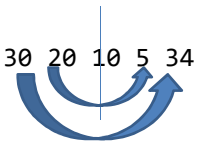
```
陣列A的內容為(2x3)：
01 02 03
04 05 06

陣列B的內容為(2x3)：
07 08 09
10 11 12

陣列A+B=C，陣列C的內容為(3x3)：
08 10 12
14 16 18
```

40

回家作業

- hw6_4.java

請撰寫程式, 將陣列的內容反轉, 舉例來說, 如果陣列的內容如下：
30,20,10,5,34 您的程式必須將陣列內容改為：34,5,10,20,30



比較次數是數列的一半

(1) 先把30放到temp
再把34放到30的位置
接著把temp的值(30)放到34的位置

(2) 先把20放到temp
再把5放到20的位置
接著把temp的值(20)放到5的位置

.....依序比較

原始陣列內容：
30 20 10 5 34
反轉之後內容：
34 5 10 20 30

41

回家作業

- hw6_5.java [泡泡排序法]

1. 請使用泡泡排序法(Bubble Sort)撰寫程式。
2. 程式內有一鍵盤輸入的資料陣列。
3. 請輸出泡泡排序法(由大排到小)的比對過程。
4. 執行結果如下所顯示。

請輸入5個整數：3 5 2 8 7
5 3 8 7 2
5 8 7 3 2
8 7 5 3 2
8 7 5 3 2

請輸入5個整數：5 4 3 2 1
5 4 3 2 1
5 4 3 2 1
5 4 3 2 1
5 4 3 2 1

43

氣泡排序法(Bubble Sort)

【定義】

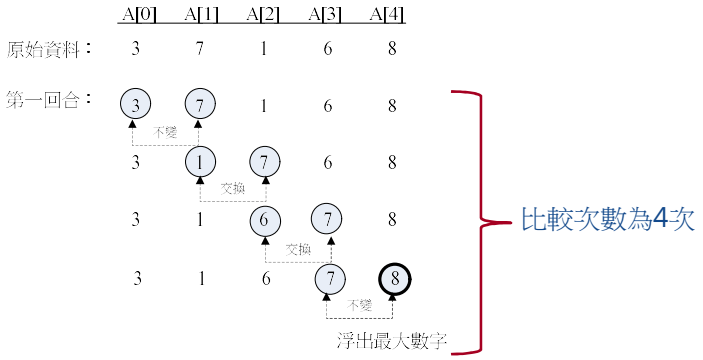
是指將兩個相鄰的資料相互做比較，若比較時發現次序不對，則將兩個資料互換，並且資料依序由上往下比，而結果則會依序由下往上浮起，猶如氣泡一般。

44

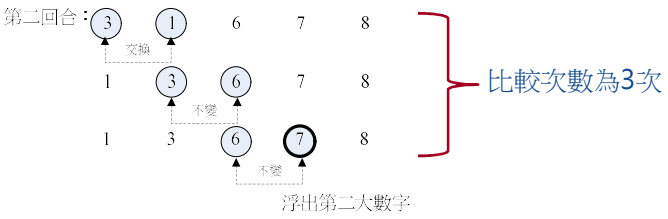
【原理】

由小到大
or
由大到小

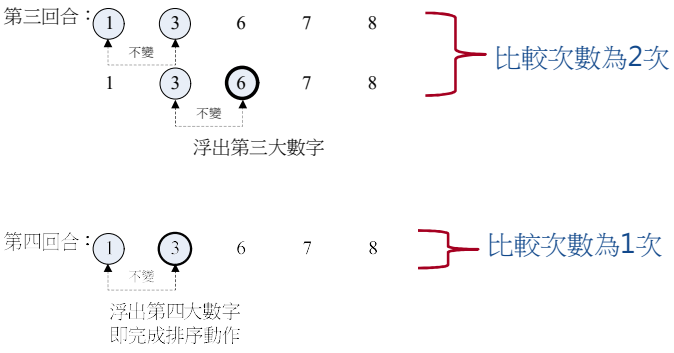
逐次比較兩個相鄰的資料，按照排序的條件交換位置，直到全部資料依序排好為止。其排序過程。如下圖所示：



45


第二回合：
3 1 6 7 8
交換
1 3 6 7 8
不變
1 3 6 7 8
不變
浮出第二大數字
比較次數為3次

46


第三回合：
1 3 6 7 8
不變
1 3 6 7 8
不變
浮出第三大數字
比較次數為2次
第四回合：
1 3 6 7 8
不變
浮出第四大數字
即完成排序動作
比較次數為1次

47

	陣列	A[0]	A[1]	A[2]	A[3]	A[4]	說明 n=5
	輸入	3	7	1	6	8	目標：1 3 6 7 8 方法：兩兩比較
共需要比較n-1輪	第1輪	37168 31768 31678 31678←第1輪比較結束，印出結果(1st出現)					第i輪，需要比較n-i次
	第2輪	1367 8 1367 8 1367 8←第2輪比較結束，印出結果(2nd出現)					
	第3輪	136 78 136 78←第3輪比較結束，印出結果(3rd出現)					
	第4輪	13 678←第4輪比較結束，印出結果(4th出現)					

49

⊕氣泡排序法的演算法如下：

```
01 Procedure BubSort(int A[], int n)
02  begin
03   for (i=n-1; i>=1; i--)            //排序n-1個回合
04    {
05      for (j =0; j <=i-1; j++)        //從第0個元素開始掃瞄
06       if (A[j] > A[j+1])             //判斷左邊元素是否大於右邊元素
07         {                            // A[j] 與 A[j+1]交換
08           Temp = A[j];
09           A[j] = A[j+1];
10           A[j+1] = Temp;
11         }
12    }
13  end
14 End Procedure
```

50

【實例】

假設原始資料為：3,7,1,6,4，在進行排序時，每一回合必定會有一個元素排到定位，稱為一個回合(Pass)。

	A[0]	A[1]	A[2]	A[3]	A[4]	比較次數	比較範圍
原始資料	3	7	1	6	4		
Pass 1	3	1	6	4	**7**	4	(A[0]與A[1]、A[1]與A[2]、A[2]與A[3]、A[3]與A[4])
Pass 2	1	3	4	**6**	7	3	(A[0]與A[1]、A[1]與A[2]、A[2]與A[3])
Pass 3	1	3	**4**	6	7	2	(A[0]與A[1]、A[1]與A[2])
Pass 4	1	**3**	**4**	6	7	1	(A[0]與A[1])
					總比較次數	10	

51

【分析】

1. 比較之回合次數=資料個數-1

   (例如：資料個數n=5，則回合次數為4)

2. 在每一回合之後，至少會有一個資料可以排列到正確位置，再進行下一個回合的排列時，便可以減少此資料的比較。

   (例如：資料個數n=5，則Pass 1時，比較次數為4，Pass 2時，比較次數為3，以此類推，如上表所示)

3. 需要一個額外空間。

   例如：在上面的演算法中的行號08，需要一個Temp變數空間。

4. 為一種穩定排序 (Stable Sorting) 。

   因為氣泡排序法交換條件為「左大右小」時才必須交換。如下所示：

(1)排序前：

3	7	1	6	4

(2)排序後：

1	3	4	6	7

52

回家作業

- hw6_6.java [停車費計算]

1. 請用陣列方式寫出停車費用計算的程式。
2. 假設停車時段分為：
   a. 2小時以內(含2小時)，每小時以30元計算。
   b. 2小時以上不足4小時，每小時以50元計算。 c. 4小時以上不足6小時，每小時以80元計算。 d. 6小時以上，每小時以100元計算。
3. 請輸入停車時數並計算出停車費用，分別計算2小時、3小時、5小時及8小時的應繳費用，顯示結果如下 。

提示：
hour= {0, 2, 4, 6}; // 時段
fee= {30, 50, 80, 100}; // 時段費率

```
停車時數：2小時
應繳費用：60元整
--------------------
停車時數：3小時
應繳費用：110元整
--------------------
停車時數：5小時
應繳費用：240元整
--------------------
停車時數：8小時
應繳費用：520元整
```

53

回家作業

- hw6_7.java [選擇排序法]

1. 請使用選擇排序法(Selection Sort)撰寫程式。
2. 程式內有一資料陣列{1, 3, 2, 5, 4, 6}。
3. 請輸出選擇排序法的比對過程。
4. 執行結果如下頁所顯示。

```
1 3 2 5 4 6
1 2 3 5 4 6
1 2 3 5 4 6
1 2 3 4 5 6
1 2 3 4 5 6
```

54

選擇排序法（Selection Sort）

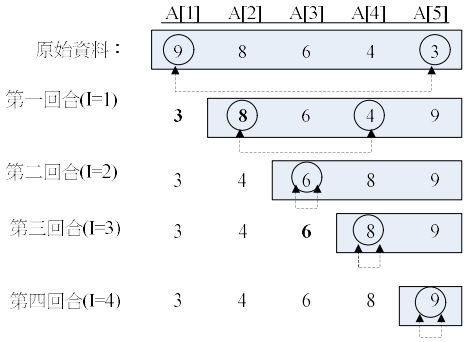
【定義】

先以某一數值為基準，再由左至右掃瞄比目前大或小的數字，找到時，先記錄其位置或索引值，待確定後再進行資料的交換，而這樣的方法我們稱之為選擇排序法（Selection Sort）。

55

【原理】

第一回合由資料中選取最小的資料和第一個資料對調、第二回合由資料中選取第二小的資料和第二個資料對調 (因最小的資料已排到第一個位置)、依此循環直到最後一個資料，即完成資料的排序。如下圖所示：



56

「選擇排序法」的演算法如下：

```
01  Procedure SelSort(int A [], int n)
02   Begin
03   for (i = 0; i < n - 1; i++)        //控制排序n-1個回合
04    {
05      Min = i;                        //設定最小值索引
06      for (j = i + 1; j <= n; j++)    //從第1個元素開始掃瞄
07        if (A[Min]>A[j])              //如果左邊元素大於右邊元素
08          Min = j;                    //則重新設定最小值索引
09     {                                //並進行兩個的資料交換位置
10      Temp = A[i];
11      A[i] = A[Min];
12      A[Min] = Temp;
13     }
14    }
15   End
16  End Procedure
```

57

回家作業

- hw6_8.java [二分排序法]

1. 程式內有已排序資料{5, 9, 13, 15, 17, 19, 25, 30, 45}，請使用二分搜尋法尋找輸入的資料。
2. 程式連續執行兩次，於程式執行時，如下頁所示，顯示【請輸入要找尋的資料：】要求輸入欲尋找的資料n。
3. 若沒有搜尋到相符的數值，顯示【n不在陣列中】，將欲尋找的資料代入n，如下頁所示。
4. 尋找時，列出尋找區間及此區間的中間值，搜尋幾次就列出幾項，最後產出【經過 y 次的尋找】， y代入搜尋次數；若有搜尋到相符的數值，值位在陣列中的第幾個位置，如下所示。

```
請輸入要找尋的資料：2
尋找區間：0(5)..8(45),中間：4(17)
尋找區間：0(5)..3(15),中間：1(9)
尋找區間：0(5)..0(5),中間：0(5)
經過 3 次的尋找
2不在陣列中
請輸入要找尋的資料：30
尋找區間：0(5)..8(45),中間：4(17)
尋找區間：5(19)..8(45),中間：6(25)
尋找區間：7(30)..8(45),中間：7(30)
經過 3 次的尋找
您要找的資料在陣列中的第7個位置
```

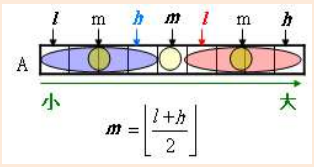
58

二分搜尋法(Binary Search)

【定義】

先將資料分割成兩部份，再利用「鍵值」與「中間值」來比較大小，如果鍵值小於中間值，則可確定要找的資料在前半段的元素中，如此分割數次直到找到為止。

【圖解】



【適用時機】事先已經排序完成。

59

【演算法1】

```
01  Procedure binsearch(A[], key , low, high)
02   Begin
03   if(low<=high)              //第一項資料的索引值小於最後一項資料的索引值
04    {
05     mid=⌊(low+high)/2⌋  ;          //計算中間值的位置
06     switch compare(key, A[mid])     //「鍵值」與「中間值」比較大小
07      {
08        Case "=": return mid;              //找到
09        Case  "<" : return binsearch(A, key, low, mid-1) ;   //遞迴呼叫找左半部
10        Case ">": return binsearch(A, key, mid+1, high) ;    //遞迴呼叫找右半部
11      }
12    }
13    Return -1;                          //搜尋失敗，傳回-1
14   End
15  End Procedure
```

60

步驟

資料需依大小先排序好

Middle = ⌊(Left + Right)/2⌋

將鍵值key與搜尋範圍的中間資料data[Middle]作比對

key = data[Middle]：找到

key < data[Middle]：縮小搜尋範圍 ⇒ Right = Middle-1

key > data[Middle]：縮小搜尋範圍 ⇒ Left = Middle+1

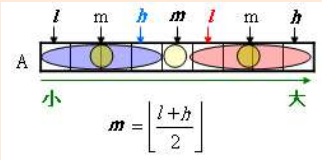
重複上步驟，直到找到資料或搜尋範圍交叉(找不到)

61

演算法2

```
function search(array[], target)
        var left = 0, right = array.length - 1
        while left <= right
                var middle = 取出array中點
                if array[middle] == target
                        return middle
                else if array [middle] > target
                        right = middle – 1 // 要找的比目標小，再搜索左半邊
                else if array[middle] < target
                        left = middle + 1 // 要找的比目標大，再搜索右半邊

                end if
        end while
        return -1
end function
```

62



步驟① 將所有資料由小至大排序。

步驟② 設L(Low)表示第一項資料(最小)的索引，

H(High)表示最後一項資料(最大)的索引。

步驟③ M(Middle)表示中間項的索引＝ $\lfloor(L+H)/2\rfloor$

步驟④ 將「鍵值 」和「中間值」做比較。

當鍵值 ＞中間值 ，則L＝M＋1。

當鍵值 ＜中間值 ，則H＝M－1。

當鍵值 ＝中間值 ，則表示已經找到。

步驟⑤重新計算M(中間值之索引)之後，再重複步驟③和⑤，

直到找到或所有資料均找過為止。

63

【實例】

假設有九筆資料：8,1,99,76,88,45,15,33,24

現在我們想從資料中尋找「88」，請利用二分搜尋法來進行搜尋，

並撰寫出完整的步驟。

【解答】

步驟① 將所有資料由小至大排序之後，並依序存入一維陣列中。

1, 8, 15, 24, 33, 45, 76, 88, 99

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
1	8	15	24	33	45	76	88	99

步驟② 設L(Low)表示第一項資料(最小) 的索引，

H(High)表示最後一項資料(最大) 的索引。

已知：L=0, H=8

64

步驟③ 計算M(Middle)表示中間值的索引= $\lfloor (L+H)/2 \rfloor$

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
1	8	15	24	33	45	76	88	99

由於L=0, H=8，所以代入公式M= $\lfloor (L+H)/2 \rfloor$

M=$\lfloor (0+8)/2 \rfloor$ =4

步驟④ 將「鍵值」和「中間值」做比較。

假設鍵值Key=88

因為，Key=88>A[4]=33，所以

當鍵值＞中間值，所以，欲尋找的資料在右半邊，

因此，重新計算L＝M＋1=4+1=5

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
1	8	15	24	33	45	76	88	99

忽略不理　　鍵值在右邊

65

步驟⑤ 重新計算M(中間值之索引)，重複步驟③和⑤，

直到找到或所有資料均找過為止。

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
1	8	15	24	33	45	76	88	99

忽略不理　　鍵值在右邊

❶M=$\lfloor (5+8)/2 \rfloor$=6

因為，Key=88>A[6]=76，所以

當鍵值＞中間值， 所以，欲尋找的資料在右半邊，

因此，重新計算L＝M＋1=6+1=7

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
1	8	15	24	33	45	76	88	99

忽略不理　忽略不理　鍵值在右邊

❷M= $\lfloor (7+8)/2 \rfloor$ =7

因為，Key=88=A[7]，所以找尋成功。

66

回家作業

hw6_9.java

由鍵盤輸入5個整數，存放到一維陣列，並計算元素中所有元素的立方和。

```
請輸入5個整數：1 2 3 4 5
原始陣列內容：
1 2 3 4 5
全部元素立方和：225
```

68

回家作業

- **hw6_10.java**
- 假設某一公司有五種產品A、B、C、D與E，其單價分別為12、16、10、14與15元；而該公司共有三位銷售員，他們在某個月份的銷售量如下所示:
- 銷貨員 產品A 產品B 產品C 產品D 產品E

1	33	32	56	45	33
2	77	33	68	45	23
3	43	55	43	67	65

- 試寫一程式印出上表的內容，並計算:

(a)每一個銷貨員的銷售總金額

(b)每一項產品的銷售總金額

(c)有最好業績(銷售總金額為最多者)的銷售員

(d)銷售總金額為最多的產品

```
產品A的總銷售額=153
產品B的總銷售額=120
產品C的總銷售額=167
產品D的總銷售額=157
產品E的總銷售額=121

賣最好的產品是第3產品    總共賣了:167元
賣最差的產品是第2產品    總共賣了:120元
```

69

回家作業

hw6_11.java

請說明以下列印相關語法的異同，並舉程式說明其差異。

1. System.out.println()
2. System.out.print()
3. System.out.print(\n)
4. System.out.print(\r\n)
5. System.out.printf()
6. System.out.format()

71