

陣列（一）

人工智慧與無線感應設備開發專班

湜憶電腦知訊顧問股份有限公司

馬傳義

前言

- 當一個資料要儲存於記憶體中，需要使用一個變數來儲存；若有100筆資料，則需要使用100個變數。
- 除了宣告變數外，程式中的敘述也變得很繁雜，維護上也很困難。
- 若這些變數的資料型別相同，而且相關聯，我們可以利用『陣列』來取代變數，以減少程式在維護時的困難度。

前言

- 陣列源自數學中的矩陣（Matrix）。
- 它是一群具有相同資料型別的變數或物件之集合。
- 陣列中每一個變數或物件稱為陣列元素（Array element），可簡稱為元素。
- 陣列元素使用相同名稱（即陣列名稱），元素之間以索引值的不同來區別。
- 陣列中只有一組索引值時，稱為一維陣列，有兩組索引值時稱為二維陣列，依此類推。

一維陣列

陣列宣告

◆ 語法：

資料型態[] 陣列名稱; ←建議使用

資料型態 []陣列名稱;

資料型態 陣列名稱[];

◆ 說明：

- 資料型態：

- ◆ 陣列中所有的資料都是此資料型態。

- []：

- ◆ 表示一維陣列。

- 陣列名稱：

- ◆ 是陣列中所有資料的共同名稱。

一維陣列

例如：

```
int[] A;
```

```
String []course;
```

```
float data[];
```

注意：

陣列宣告後，必須指定『陣列大小』後，方可使用。

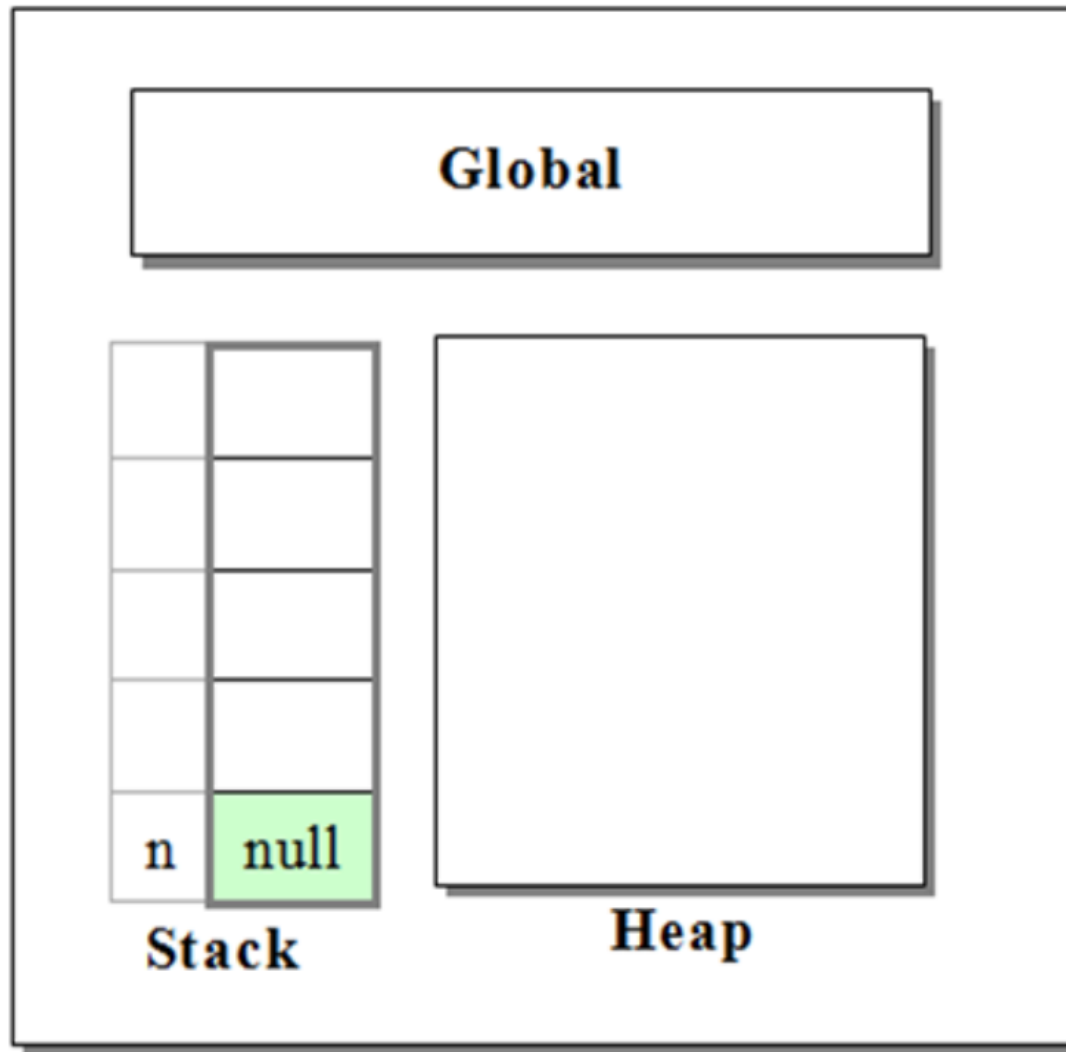
一維陣列

- ◆ 陣列宣告後，它在記憶體中，只是個「參考型別」，此時（如下圖），
 - 在Stack（堆疊）中，會配置一塊空間，初始值為「null」。
 - 在Heap（堆積）並沒有配置陣列物件實體，表示此陣列目前沒有任何資料。

例如：

```
int[] n;
```

一維陣列



一維陣列

陣列大小

- ◆ 陣列宣告後，必須產生陣列實體（即宣告陣列的大小），才能將資料存放於陣列實體中。

- ◆ 語法：

陣列名稱 = new 資料類型[陣列大小];

- ◆ 說明：

陣列大小：

代表陣列中有多少的元素。

例如：

```
age=new int[5];
```

即是對之前宣告的陣列age配置5個元素的陣列空間。

一維陣列

注意：

陣列的元素的索引值從0開始，最後一個陣列的索引值則為陣列大小減1。

例如：

上例宣告後的索引值為：age[0]~age[4]。

◆ 亦可在宣告陣列的同時，直接指定陣列的大小。

- 語法：

資料型態[] 陣列名稱 = new 資料型態[陣列大小];

例如：

```
int[] A=new int[5];
```

```
String []course=new String[5];
```

```
float data[]=new float data[5];
```

一維陣列

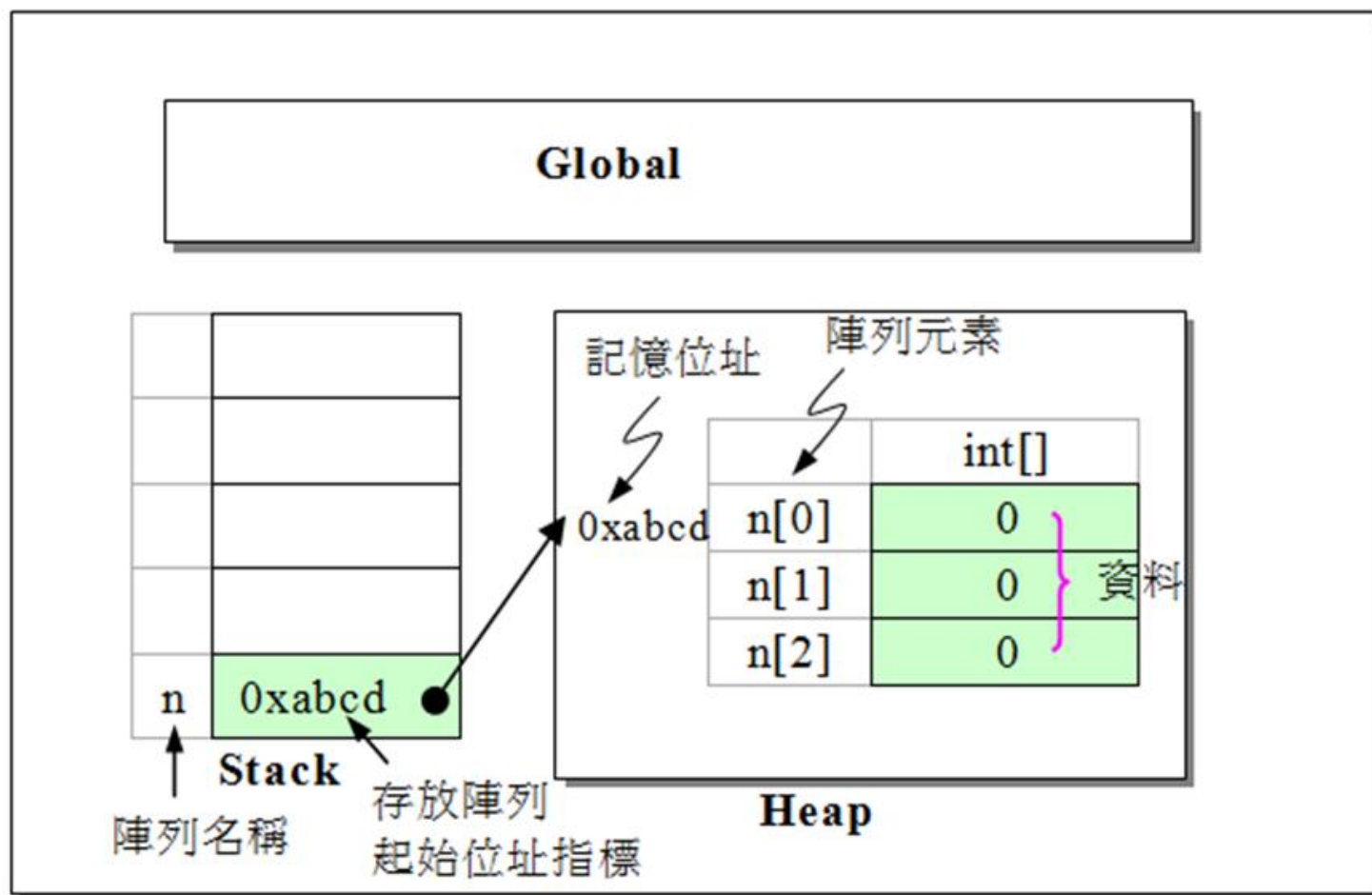
- ◆ 宣告陣列大小後，會依序存放在Heap（堆積）中的連續記憶體空間，然後在Stack（堆疊）中的陣列變數會參考（指到）陣列的第一個元素的記憶體位址（如下圖）。
- ◆ 此時，陣列中的所有元素，會依照宣告的資料型態，進行初始化。

| 陣列的資料型態 | 初始值 |
|---------|-------------|
| 數字 | 0 |
| 字元 | Unicode的字元0 |
| 布林 | false |
| 物件 | null |

一維陣列

例如：

```
n = new int[3];
```



一維陣列

注意：

- 上述陣列的記憶體配置情形，在一般的情況下，Java會自動管理。
- 程式設計者並不需要真的瞭解在Stack（堆疊）中，陣列變數參考（儲存的）陣列的第一個元素的記憶體位址；亦不需要真的知道在Heap（堆積）中的記憶體狀況。
- 程式設計者只須利用陣列的「索引值」，即可存取陣列中的元素。

例如：

```
n = new int[5];
```

| 陣列索引 | n[0] | n[1] | n[2] | n[3] | n[4] |
|------|------|------|------|------|------|
| 元素值 | 0 | 0 | 0 | 0 | 0 |

一維陣列

- ◆ 陣列中有定義一個方法，可以讓使用者取得陣列的長度（即陣列的大小）：


- 語法：

陣列名稱.length;

例如：

```
course.length;
```

一維陣列

 程式：

```
public class CH04_01
{
    public static void main(String[] args)
    {
        int[] A;
        A = new int[5];
        System.out.println("A陣列長度：" + A.length);
        System.out.print("A陣列元素值：");
        for (int i = 0; i < A.length; i++)
            System.out.print(A[i] + " ");

        System.out.println("\n");
    }
}
```

一維陣列

```
int[] B = new int[5];  
System.out.println("B陣列長度：" + B.length);  
System.out.print("B陣列元素值：");  
for (int i = 0; i < B.length; i++)  
    System.out.print(B[i] + "t");  
}  
}
```

一維陣列

```
1 public class CH04_01
2 {
3     public static void main(String[] args)
4     {
5         int[] A;
6         A = new int[5];
7         System.out.println("A陣列長度：" + A.length);
8         System.out.print("A陣列元素值：");
9         for (int i = 0; i < A.length; i++)
10             System.out.print(A[i] + "\t");
11
12         System.out.println("\n");
13
14         int[] B = new int[5];
15         System.out.println("B陣列長度：" + B.length);
16         System.out.print("B陣列元素值：");
17         for (int i = 0; i < B.length; i++)
18             System.out.print(B[i] + "\t");
19     }
20 }
```


一維陣列

◆ 執行結果：

```
A陣列長度：5  
A陣列元素值：0 0 0 0 0  
  
B陣列長度：5  
B陣列元素值：0 0 0 0 0
```

◆ 說明：

- 行05：
 - ◆ 宣告一維整數陣列A。
- 行06：
 - ◆ 指定A陣列的大小。
- 行07：
 - ◆ 螢幕輸出陣列的長度（即陣列的大小）。

一維陣列

- 行08：
 - ◆ 螢幕輸出。
- 行09~行10：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
- 行12：
 - ◆ 螢幕輸出。
- 行14：
 - ◆ 宣告一維整數陣列B，並同時指定陣列B的大小。
- 行15：
 - ◆ 螢幕輸出陣列的長度（即陣列的大小）。
- 行16：
 - ◆ 螢幕輸出。

一維陣列

- 行17~行18：
 - ◆ 利用for迴圈，輸出B陣列中各個元素的值。

一維陣列

◆ 課堂練習：

- 請將此（CH04_01）程式，改用for-each迴圈，顯示陣列中各個元素的值。

一維陣列

指定初值

- ◆ 在Java中，必須給予陣列初始值後才能對陣列進行操作。
- ◆ 預設初值：
 - Java語言在陣列產生時會針對各資料型態預設初始值，以下為各資料型態的預設初始值：

| 陣列的資料型態 | 初始值 |
|---------|-------------|
| 數字 | 0 |
| 字元 | Unicode的字元0 |
| 布林 | false |
| 物件 | null |

一維陣列

◆ 指定初值：

- 語法1（個別設定初值）：

陣列名稱[陣列索引值]=初值;

- 說明：

- ◆ 陣列索引值：

- ◆ 陣列的位置。

注意：

陣列的索引值是從『0』開始算起。

例如：

A[0]=20;

A[1]=30;

A[2]=40;

一維陣列

- 語法2（宣告陣列時，同時設定大小與初值）：

資料型態[] 陣列名稱=new 資料型態[] {初值1,初值2,...};

- 說明：

- ◆ 陣列中設定初始值時，需要用大括號（{}）和逗號（,）來分隔。

例如：

```
int[] A=new int[]{ 10,20,30,40,50};
```

注意：

陣列的大小是指陣列中有多少『元素』，而陣列『索引值』是指某元素在陣列中第幾個『位置』。

一維陣列

◆ 程式：

```
public class CH04_02
{
    public static void main(String[] args)
    {
        String[] course = new String[5];

        course[0] = "姓名";
        course[1] = "國文";
        course[2] = "數學";
        course[3] = "社會";
        course[4] = "自然";

        for (int i = 0; i < course.length; i++)
            System.out.print(course[i] + " ");
    }
}
```


一維陣列

```
System.out.println();
```

```
System.out.print("吳勁律\t");
```

```
int[] score = new int[] { 100, 96, 97, 86 };
```

```
int sum = 0;
```

```
for (int i = 0; i < score.length; i++)
```

```
{
```

```
    System.out.print(score[i] + "\t\t");
```

```
    sum += score[i];
```

```
}
```

```
System.out.println();
```

```
System.out.println("\n總分=" + sum);
```

```
System.out.println("\n平均=" + (float) sum / score.length);
```

```
}
```

```
}
```

一維陣列

```
1 public class CH04_02
2 {
3     public static void main(String[] args)
4     {
5         String[] course = new String[5];
6
7         course[0] = "姓名";
8         course[1] = "國文";
9         course[2] = "數學";
10        course[3] = "社會";
11        course[4] = "自然";
12
13        for (int i = 0; i < course.length; i++)
14            System.out.print(course[i] + "\t");
15
16        System.out.println();
17        System.out.print("吳勁律\t");
18
19        int[] score = new int[] { 100, 96, 97, 86 };
20        int sum = 0;
21        for (int i = 0; i < score.length; i++)
22        {
23            System.out.print(score[i] + "\t\t");
24            sum += score[i];
25        }
26        System.out.println();
```

一維陣列

```
27     System.out.println("\n總分=" + sum);  
28     System.out.println("\n平均=" + (float) sum / score.length);  
29 }  
30 }
```

◆ 執行結果：

| 姓名 | 國文 | 數學 | 社會 | 自然 |
|----------|-----|----|----|----|
| 吳勁律 | 100 | 96 | 97 | 86 |
| 總分=379 | | | | |
| 平均=94.75 | | | | |

◆ 說明：

- 行05：
 - ◆ 宣告一維字串陣列course，並同時指定大小。
- 行07~行11：
 - ◆ 以個別指定方式，給予陣列course初值。

一維陣列

- 行13~行14：
 - ◆ 利用for迴圈，輸出course陣列中各個元素的值。
- 行16：
 - ◆ 螢幕輸出。
- 行17：
 - ◆ 螢幕輸出。
- 行19：
 - ◆ 宣告一維整數陣列score，並同時指定陣列的初值。
 - ◆ 因為直接指定陣列的初值，故不需另外再指定score陣列的大小。
- 行20：
 - ◆ 宣告變數，並指定初值。

一維陣列

- 行21~行25：
 - ◆ 利用for迴圈，輸出score陣列中各個元素的值，並計算各元素值的總和。
 - ◆ 行21：
設定for迴圈。
 - ◆ 行23：
輸出score陣列中各個元素的值。
 - ◆ 行24：
計算score陣列中各個元素值的總和。
- 行26~行28：
 - 螢幕輸出。

一維陣列

◆ 課堂練習：

- 請將此（CH04_02）程式更改，以符合下列條件：
 - 1、指定『科目』初值的部分，與陣列宣告合併成一行。
 - 2、指定『成績』初值的部分，改由『使用者輸入』，並檢查使用者輸入的成績，必須介於0~100間。

二維陣列

陣列宣告

◆ 語法：

資料型態[][] 陣列名稱 = new 資料型態[列大小][欄大小];

◆ 說明：

- 資料型態：
 - ◆ 陣列中所有的資料都是此資料型態。
- [][]：
 - ◆ 表示二維陣列。
- 陣列名稱：
 - ◆ 是陣列中所有資料的共同名稱。
- 列大小：
 - ◆ 表示陣列第一維的長度。

二維陣列

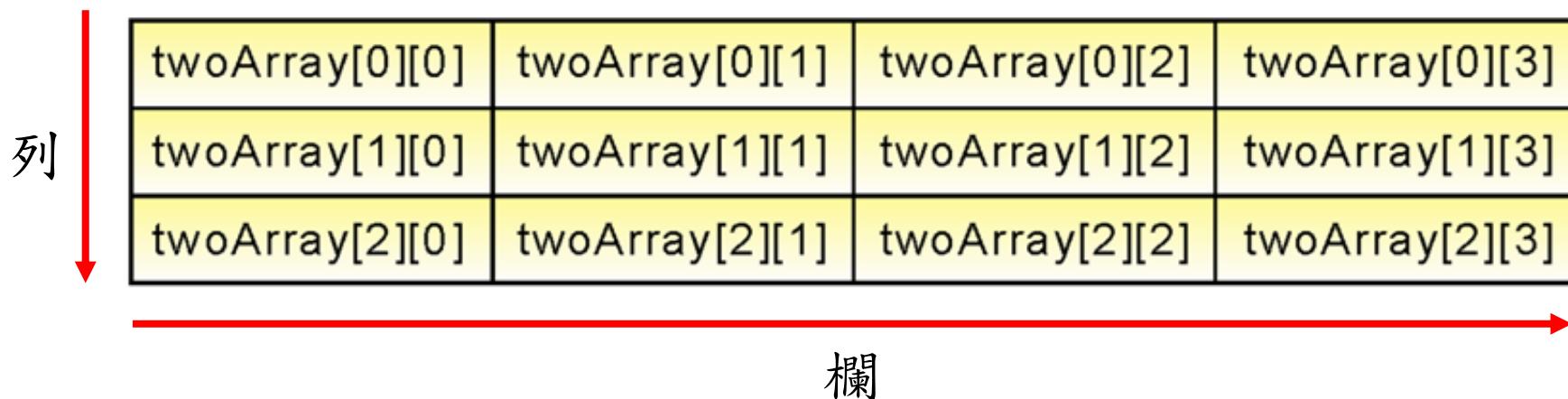
- 行大小：
 - ◆ 表示陣列第二維的長度。

例如：

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

- 說明：

假設宣告一個「3列*4欄」的（3×4）整數陣列，則每個元素的索引值如下圖：



The diagram illustrates a 3x4 2D array. A vertical red arrow on the left points downwards and is labeled '列' (Column). A horizontal red arrow at the bottom points to the right and is labeled '欄' (Row). The array is represented as a table with 3 rows and 4 columns. Each cell contains a Java-style array access notation.

| | | | |
|----------------|----------------|----------------|----------------|
| twoArray[0][0] | twoArray[0][1] | twoArray[0][2] | twoArray[0][3] |
| twoArray[1][0] | twoArray[1][1] | twoArray[1][2] | twoArray[1][3] |
| twoArray[2][0] | twoArray[2][1] | twoArray[2][2] | twoArray[2][3] |

二維陣列

注意：

與一維陣列相同，二維陣列也可以將宣告與指定陣列大小分成兩行敘述。

二維陣列

指定初值

◆ 語法：

```
int[][] twoArray=new int[][]  
{  
    {初值[0][0],初值[0][1],...},  
    {初值[1][0],初值[1][1],...},  
    {初值[2][0],初值[2][1],...},  
    ⋮  
};
```

二維陣列

◆ 說明：

- 在二維陣列設定初始值時，為了區隔列與欄，
 - ◆ 須以大括號（{ }）括住每一欄（內層）的元素初始值，並以逗號（,）區隔每個欄元素，
 - ◆ 再以大括號（{ }）括住每一列（外層）的元素初始值，並以逗號（,）區隔每個列的元素。

例如：

```
int[][] twoArray=new int[][]{{10,20,30},{40,50,60},  
                               {70,80,90}};
```

注意：

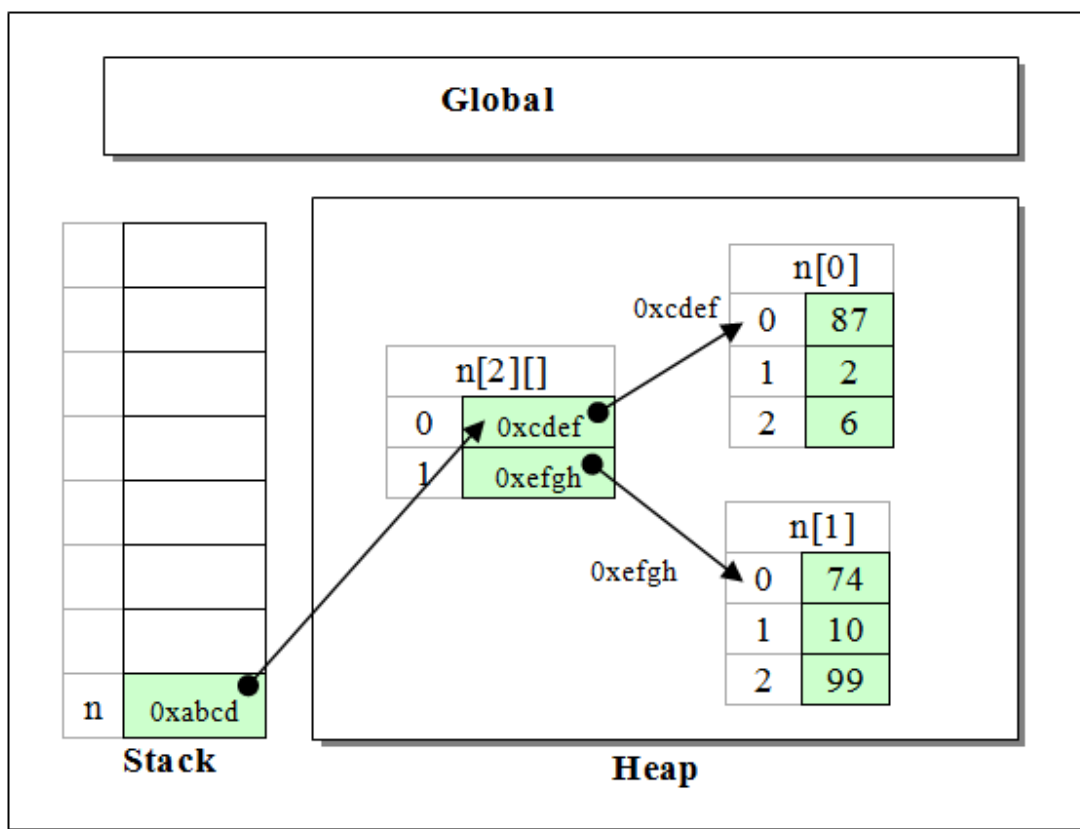
與一維陣列相同，二維陣列也可以個別設定初值。

二維陣列

◆ 二維陣列的記憶體配置情形：

例如：

```
int[][] n={{87,2,6},{74,10,99}};
```



二維陣列

◆ 程式：

```
public class CH04_03
{
    public static void main(String[] args)
    {

        String[] arr1 = new String[]
            { "座號", "國文", "英文", "數學", "最高分", "最低分" };

        int[][] arr2 = new int[][]
            { { 1, 92, 88, 76 }, { 2, 90, 98, 70 }, { 3, 82, 69, 98 } };

        for (int r = 0; r < arr1.length; r++)
            System.out.print(arr1[r] + "\t");

        System.out.println();

        int max = 0, min = 100;
```

二維陣列

```
for (int i = 0; i < arr2.length; i++)
{
    for (int j = 0; j < arr2[i].length; j++)
    {
        if (arr2[i][j] > max)
            max = arr2[i][j];

        if (j > 0)
            if (arr2[i][j] < min)
                min = arr2[i][j];

        System.out.print(arr2[i][j] + " ");
    }
    System.out.print(max + " " + min);
    System.out.println();
}
```

二維陣列

```
1 public class CH04_03
2 {
3     public static void main(String[] args)
4     {
5
6         String[] arr1 = new String[] { "座號", "國文", "英文", "數學", "最高分", "最低分" };
7
8         int[][] arr2 = new int[][] { { 1, 92, 88, 76 }, { 2, 90, 98, 70 }, { 3, 82, 69, 98 } };
9
10        for (int r = 0; r < arr1.length; r++)
11            System.out.print(arr1[r] + "\t");
12
13        System.out.println();
14
15        int max = 0, min = 100;
16
17        for (int i = 0; i < arr2.length; i++)
18        {
19            for (int j = 0; j < arr2[i].length; j++)
20            {
21                if (arr2[i][j] > max)
22                    max = arr2[i][j];
23
24                if (j > 0)
25                    if (arr2[i][j] < min)
26                        min = arr2[i][j];
27            }
28        }
29    }
30 }
```

二維陣列

```
27
28     System.out.print(arr2[i][j] + "\t\t");
29 }
30 System.out.print(max + "\t\t" + min);
31 System.out.println();
32 }
33 }
34 }
```

◆ 執行結果：

| 座號 | 國文 | 英文 | 數學 | 最高分 | 最低分 |
|----|----|----|----|-----|-----|
| 1 | 92 | 88 | 76 | 92 | 76 |
| 2 | 90 | 98 | 70 | 98 | 70 |
| 3 | 82 | 69 | 98 | 98 | 69 |

◆ 說明：

• 行06：

- ◆ 宣告一維字串陣列arr1，並同時指定初值。

二維陣列

- 行08：
 - ◆ 宣告二維整數陣列arr2，並同時指定初值。
- 行10~行11：
 - ◆ 利用for迴圈，輸出arr1陣列中各個元素的值。
- 行13：
 - ◆ 螢幕輸出。
- 行15：
 - ◆ 宣告變數，並指定初值。
- 行17~行32：
 - ◆ 利用巢狀for迴圈及if、巢狀if，輸出arr2陣列中各個元素的值，並找出最高分與最低分分數。
 - ◆ 行17：
設定外層for迴圈。
第一維的長度為3（即 `arr2.length=3`）。

二維陣列

行19：

設定內層for迴圈。

第二維的長度為4（即 `arr2[i].length=4`）。

行21~行22：

是`arr2[i][j]`中，元素的值比變數`max`中的值大，則將`arr2[i][j]`中元素的值，寫入變數`max`中。

行24~行26：

外層if是跳過`arr2[i][0]`中元素的值（此值為「座號」不是成績），不用比大小。

內層if是判斷`arr2[i][j]`中元素的值，是否比變數`min`中的值小，若是，則將`arr2[i][j]`中元素的值，寫入變數`min`中。

行28：

螢幕輸出。

二維陣列

行30~行31：

螢幕輸出。

注意：

陣列的總長度必須先取得每個維度的長度，加總之後才是陣列的總長度。

三維陣列

陣列宣告

◆ 語法：

資料型態[][][] 陣列名稱=new 資料型態[x方向][y方向][z方向];

◆ 說明：

- 資料型態：
 - ◆ 陣列中所有的資料都是此資料型態。
- [][][]：
 - ◆ 表示三維陣列。
- 陣列名稱：
 - ◆ 是陣列中所有資料的共同名稱。
- [x方向]：
 - ◆ 表示陣列第一維的長度。

三維陣列

- [y方向]：
 - ◆ 表示陣列第二維的長度。
- [z方向]：
 - ◆ 表示陣列第三維的長度。

例如：

```
int[][][] threeArray=new int[2][3][4];
```

- 說明：

假設宣告一個「 $2 \times 3 \times 4$ 」的整數陣列，則每個元素的索引值如下圖：

每個元素的索引值



三維陣列

注意：

與一、二維陣列相同，三維陣列也可以將宣告與指定陣列大小分成兩行敘述。

三維陣列

指定初值

◆ 語法：

```
int[][][] threeArray=new int[][][]  
{  
    {  
        {初值[0][0][0],初值[0][0][1],...},  
        {初值[0][1][0],初值[0][1][1],...},  
        {初值[0][2][0],初值[0][2][1],...},  
        ⋮  
    },  
    {  
        {初值[1][0][0],初值[1][0][1],...},  
        {初值[1][1][0],初值[1][1][1],...},  
        {初值[1][2][0],初值[1][2][1],...},  
        ⋮  
    }  
};
```


三維陣列

◆ 說明：

- 在三維陣列設定初始值時，為了區隔x方向、y方向與z方向，
 - ◆ 須以大括號（{ }）括住每一個z方向（內層）的元素初始值，並以逗號（,）區隔每個z方向元素，
 - ◆ 再以大括號（{ }）括住每一個y方向（中層）的元素初始值，並以逗號（,）區隔每個y方向的元素，
 - ◆ 最後再以大括號（{ }）括住每一個x方向（外層）的元素初始值，並以逗號（,）區隔每個x方向的元素。

例如：

```
int[][][] threeArray=new int[][][]{{{12,92,88,76},  
                                     {23,90,98,70},{33,82,69,98}}},{  
                                     {32,32,86,36},  
                                     {43,30,38,40},{73,92,89,28}}};
```

三維陣列

注意：

與一、二維陣列相同，三維陣列也可以個別設定初值。

三維陣列

◆ 程式：

```
public class CH04_04
{
    public static void main(String[] args)
    {
        String[][][] threeArr=new String[][][]{
            {
                {"000","001","002","003"},
                {"010","011","012","013"},
                {"020","021","022","023"}
            },
            {
                {"100","101","102","103"},
                {"110","111","112","113"},
                {"120","121","122","123"}
            }
        };
    }
}
```

三維陣列

```
System.out.println("三維陣列輸出結果：");
```

```
for (int i = 0; i < threeArr.length; i++)  
{  
    for (int j = 0; j < threeArr[i].length; j++)  
    {  
        for(int k = 0; k<threeArr[i][j].length; k++)  
            System.out.print(threeArr[i][j][k] + "\t\t");  
        System.out.println();  
    }  
    System.out.println();  
}
```

```
System.out.println("隨機挑選三維陣列元素");  
System.out.println("threeArr[0][0][0]="+threeArr[0][0][0]);  
System.out.println("threeArr[0][1][2]="+threeArr[0][1][2]);  
System.out.println("threeArr[1][0][1]="+threeArr[1][0][1]);  
System.out.println("threeArr[1][1][2]="+threeArr[1][1][2]);  
}
```

三維陣列

```
1 public class CH04_04
2 {
3     public static void main(String[] args)
4     {
5         String[][][] threeArr = new String[][][] {
6             {
7                 { "000", "001", "002", "003" },
8                 { "010", "011", "012", "013" },
9                 { "020", "021", "022", "023" }
10            },
11            {
12                { "100", "101", "102", "103" },
13                { "110", "111", "112", "113" },
14                { "120", "121", "122", "123" }
15            }
16        };
17
18        System.out.println("三維陣列輸出結果：");
19
20        for (int i = 0; i < threeArr.length; i++)
21        {
22            for (int j = 0; j < threeArr[i].length; j++)
23            {
24                for (int k = 0; k < threeArr[i][j].length; k++)
25                    System.out.print(threeArr[i][j][k] + "\t");
26                System.out.println();
27            }
28        }
29    }
30 }
```

三維陣列

```
27     }  
28     System.out.println();  
29 }  
30  
31 System.out.println("隨機挑選三維陣列元素");  
32 System.out.println("threeArr[0][0][0]=" + threeArr[0][0][0]);  
33 System.out.println("threeArr[0][1][2]=" + threeArr[0][1][2]);  
34 System.out.println("threeArr[1][0][1]=" + threeArr[1][0][1]);  
35 System.out.println("threeArr[1][1][2]=" + threeArr[1][1][2]);  
36 }  
37 }
```

三維陣列

◆ 執行結果：

三維陣列輸出結果：

| | | | |
|-----|-----|-----|-----|
| 000 | 001 | 002 | 003 |
| 010 | 011 | 012 | 013 |
| 020 | 021 | 022 | 023 |

| | | | |
|-----|-----|-----|-----|
| 100 | 101 | 102 | 103 |
| 110 | 111 | 112 | 113 |
| 120 | 121 | 122 | 123 |

隨機挑選三維陣列元素

threeArr[0][0][0]=000

threeArr[0][1][2]=012

threeArr[1][0][1]=101

threeArr[1][1][2]=112

◆ 說明：

- 行05~行16：

- ◆ 宣告三維整數陣列threeArr，並同時指定初值。

三維陣列

- 行18：
 - ◆ 螢幕輸出。
- 行20~行29：
 - ◆ 利用巢狀for迴圈，輸出threeArr陣列中各個元素的值。
 - ◆ 行20：
設定外層for迴圈。
第一維的長度為2（即 `threeArr.length=2`）。
 - ◆ 行22：
設定中層for迴圈。
第二維的長度為3（即 `threeArr[i].length=3`）。
 - ◆ 行24：
設定內層for迴圈。
第三維的長度為4（即 `threeArr[i][j].length=4`）。

三維陣列

- ◆ 行25：
螢幕輸出。
- ◆ 行26：
螢幕輸出。
- ◆ 行28：
螢幕輸出。
- 行31~行35：
 - ◆ 螢幕輸出。

不規則陣列

■ 若遇到每一列有不同的行數時，若以最大行數來設計，就會白白浪費掉記憶空間。

例如：

每個人的個人經歷差異性很大，使用規則的陣列來存放個人經歷，就非常浪費記憶體。

■ Java允許建立不規則陣列，也就是說，每一列的行數可以視需要而不同。

不規則陣列

陣列宣告

◆ 語法：

資料型態[][] 陣列名稱 = new 資料型態[][]

```
{  
    {初值[0][0],初值[0][1],初值[0][2],初值[0][3],...},  
    {初值[1][0],初值[1][1],初值[1][2],初值[1][3],初值[1][4],...},  
    {初值[2][0],初值[2][1],初值[2][2],...},  
    ⋮  
};
```

◆ 說明：

- 所宣告的陣列，每一列的元素個數（陣列長度）不一致。
- 這種不規則的陣列宣告語法也是一種合法的宣告方式。

不規則陣列

例如：

```
int[][] twoArray=new int[][]{  
    {15,48,44,11},  
    {12,78,56,49,58},  
    {55,24,31}  
};
```

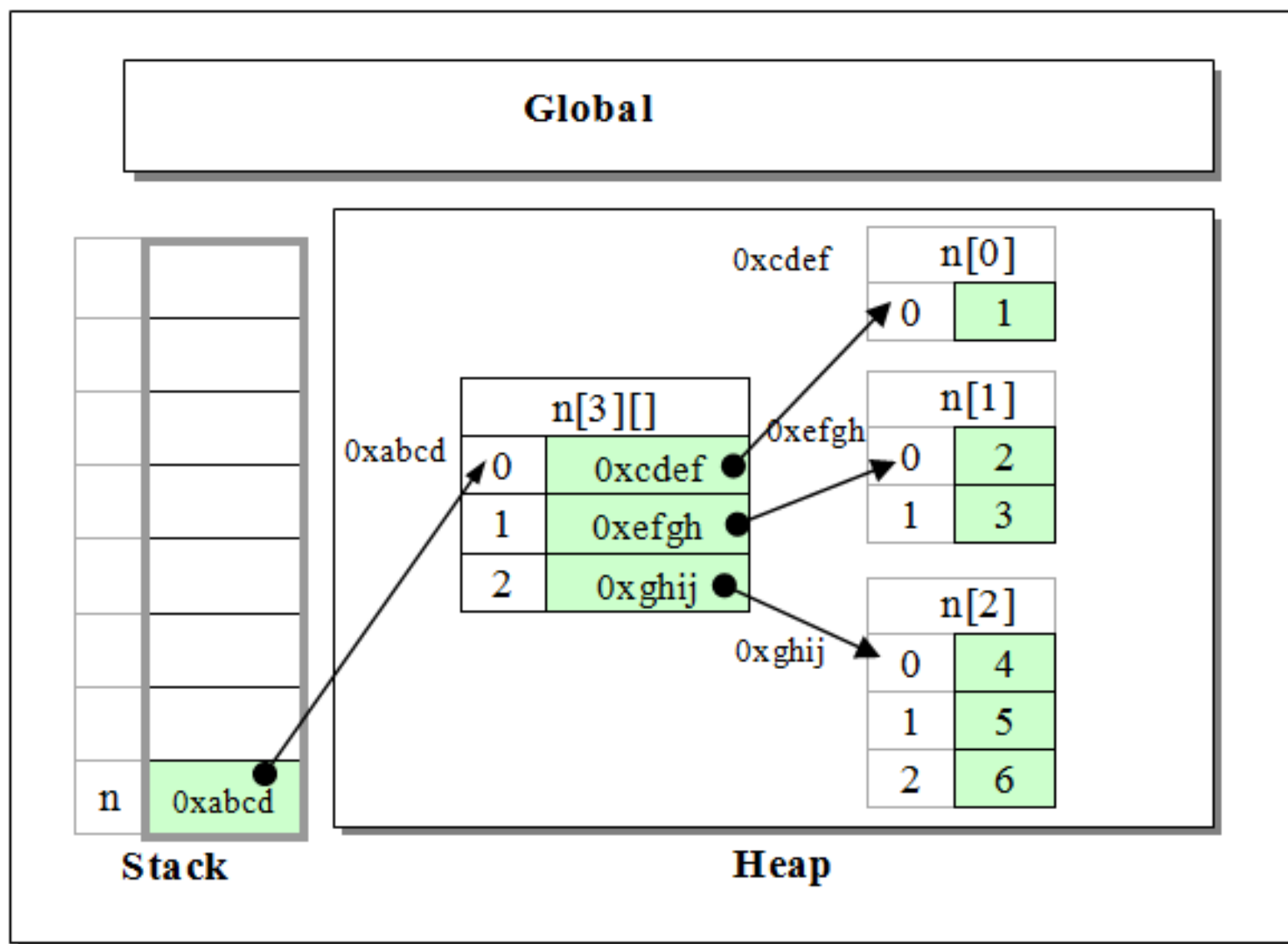
不規則陣列

◆ 不規則陣列的記憶體配置情形：

例如（如下圖）：

```
int[][] n=new int[][]{  
    {15,48,44,11},  
    {12,78,56,49,58},  
    {55,24,31}  
};
```

不規則陣列



不規則陣列

◆ 程式

```
public class CH04_05
{
    public static void main(String[] args)
    {
        String[][] data=new String[][]{
            {"王正義","校長","義工"},
            {"宋麗美","家管"},
            {"黃哲夫","律師","議員","市長"}
        };

        for(i=0;i<data.length;i++)
        {
            System.out.print(data[i][0]+" : ");
```

不規則陣列

```
for(j=1;j<data[i].length;j++)  
    if(j==data[i].length-1)  
        System.out.print(data[i][j]);  
    else  
        System.out.print(data[i][j] + " , ");  
  
    System.out.println();  
}  
}  
}
```


不規則陣列

```
1 public class CH04_05
2 {
3     public static void main(String[] args)
4     {
5         String[][] data = new String[][] {
6             { "王正義", "校長", "義工" },
7             { "宋麗美", "家管" },
8             { "黃哲夫", "律師", "議員", "市長" }
9         };
10
11         for (int i = 0; i < data.length; i++)
12         {
13             System.out.print(data[i][0] + " : ");
14
15             for (int j = 1; j < data[i].length; j++)
16                 if (j == data[i].length - 1)
17                     System.out.print(data[i][j]);
18                 else
19                     System.out.print(data[i][j] + " , ");
20
21             System.out.println();
22         }
23     }
24 }
```

不規則陣列

◆ 執行結果：

王正義：校長，義工
宋麗美：家管
黃哲夫：律師，議員，市長

◆ 說明：

- 行05~行09：

- ◆ 宣告二維整數不規則陣列data，並同時指定初值。

注意：

各列的元素值不相同。

- 行11~行22：

- ◆ 利用巢狀for迴圈及if-else，data陣列中各個元素的值。

- ◆ 行11：

設定外層for迴圈。

第一維的長度為3（即 `data.length=3`）。

不規則陣列

- ◆ 行13：

螢幕輸出每列的第一欄資料（即 姓名）。

- ◆ 行15：

設定內層for迴圈。

第二維的長度不同，分別為3、2、4
（即 `data[0].length=3`、`data[1].length=2`、
`data[2].length=4` ）。

- ◆ 行16：

判斷是否為該列的最後一筆資料。

- ◆ 行17：

若是該列的最後一筆資料，則螢幕輸出時，不要輸出逗號（，）。

- ◆ 行19：

若不是該列的最後一筆資料，則螢幕輸出時，要輸出逗號（，）。

不規則陣列

- ◆ 行21：
螢幕輸出。

重新配置陣列

- 當陣列中的元素，不再使用時，可以使用原陣列的名稱，重新配置陣列的大小，以便重新使用該陣列名稱。
- 重新配置的新陣列並不會保留原先的值，而是產生一個含有預設初值的新陣列。

◆ 語法：

陣列名稱 = new 資料類型[陣列大小];

例如：

A=new int[5];

注意：

也可以用個別重新設定初值的方法。

重新配置陣列

◆ 程式：

```
public class CH04_06
{
    public static void main(String[] args)
    {
        int[] A = new int[] { 2, 4, 6, 8, 10, 12 };
        System.out.println("顯示A陣列的元素內容：");
        for (int i = 0; i < A.length; i++)
            System.out.print(A[i] + " ");

        System.out.println("\n");

        A = new int[A.length + 1];
        System.out.println("顯示重新配置A陣列長度後的元素內容：");
        for (int i = 0; i < A.length; i++)
            System.out.print(A[i] + " ");
```

重新配置陣列

```
System.out.println("\n");
```

```
A = new int[] { 1, 3, 5, 7, 9 };
```

```
System.out.println("顯示重新配置A陣列初值後的元素內容：");
```

```
for (int i = 0; i < A.length; i++)
```

```
    System.out.print(A[i] + "t");
```

```
}
```

```
}
```

重新配置陣列

```
1 public class CH04_06
2 {
3     public static void main(String[] args)
4     {
5         int[] A = new int[] { 2, 4, 6, 8, 10, 12 };
6         System.out.println("顯示A陣列的元素內容：");
7         for (int i = 0; i < A.length; i++)
8             System.out.print(A[i] + "\t");
9
10        System.out.println("\n");
11
12        A = new int[A.length + 1];
13        System.out.println("顯示重新配置A陣列長度後的元素內容：");
14        for (int i = 0; i < A.length; i++)
15            System.out.print(A[i] + "\t");
16
17        System.out.println("\n");
18
19        A = new int[] { 1, 3, 5, 7, 9 };
20        System.out.println("顯示重新配置A陣列初值後的元素內容：");
21        for (int i = 0; i < A.length; i++)
22            System.out.print(A[i] + "\t");
23    }
24 }
```


重新配置陣列

◆ 執行結果：

顯示A陣列的元素內容：

2 4 6 8 10 12

顯示重新配置A陣列長度後的元素內容：

0 0 0 0 0 0 0

顯示重新配置A陣列初值後的元素內容：

1 3 5 7 9

◆ 說明：

- 行05：

- ◆ 宣告一維整數陣列A，並同時指定陣列的初值。

- 行06：

- ◆ 螢幕輸出。

重新配置陣列

- 行07~行08：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `A.length=6`）。
- 行10：
 - ◆ 螢幕輸出。
- 行12：
 - ◆ 重新配置A陣列的大小。
- 行13：
 - ◆ 螢幕輸出。
- 行14~行15：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為7（即 `A.length=7`）。

重新配置陣列

- 行17：
 - ◆ 螢幕輸出。
- 行19：
 - ◆ 重新配置設定A陣列中的元素值。
- 行20：
 - ◆ 螢幕輸出。
- 行21~行22：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為5（即 `A.length=5`）。

複製陣列

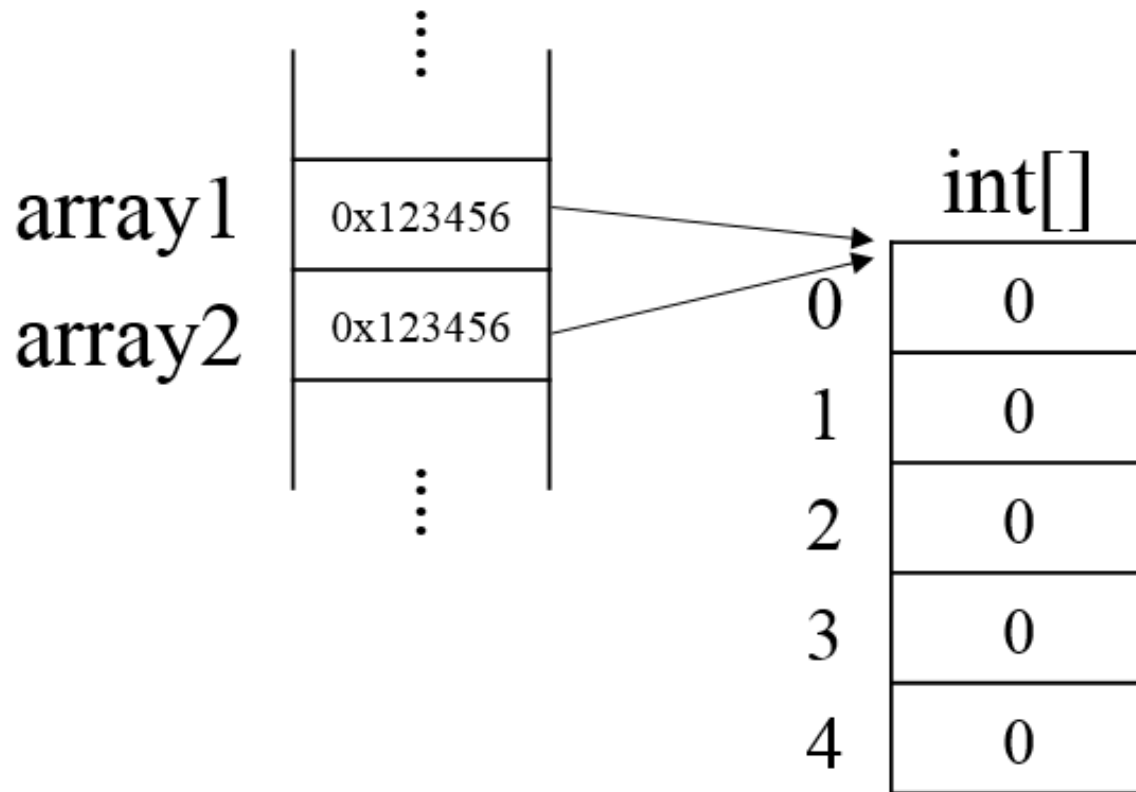
- 在JAVA中，陣列是一種參考型態，所以在陣列的複製上，不能像一般資料型態，直接使用名稱指定的方式。

例如：

```
int[] array2 = array1;
```

- 若使用上述方式，只是將array2的參考位置指向與array1相同的位置（即記憶體位置指向與array1相同的位置），如下圖。
- 也就說，當array1中的元素值被改變，array2的元素值也會跟著被改變。
- 所以用上述的方法，並沒有實際複製array1。

複製陣列



```
int[] array2 = array1;
```

記憶體配置狀態

複製陣列

■ 陣列在Java語言中有三種複製的方式：

1、迴圈複製方式：

- ◆ 當要指定一個陣列的元素值給另一個陣列時，必須分別指定兩個陣列的索引值，我們可以使用迴圈來達成這樣的複製方式。

例如：

```
int[] arr1=new int[]{1,2,3,4,5};  
int[] arr2=new int[5];  
for(int i=0; i<arr1.length; i++)  
    arr2[i]=arr1[i];
```

複製陣列

◆ 程式：

```
public class CH04_07
{
    public static void main(String[] args)
    {
        int[] A = new int[] { 2, 4, 6, 8, 10, 12 };
        System.out.println("顯示A陣列的元素內容：");
        for (int i = 0; i < A.length; i++)
            System.out.print(A[i] + " ");

        System.out.println("\n\n複製A陣列的元素值到B陣列中...\n");

        int[] B = new int[A.length];
        for (int i = 0; i < B.length; i++)
            B[i] = A[i];

        System.out.println("顯示B陣列的元素內容：");
        for (int i = 0; i < B.length; i++)
            System.out.print(B[i] + " ");
```

複製陣列

```
System.out.println("\n\n更改陣列A[1]的元素值中...\n");  
A[1] = 2;
```

```
System.out.println("重新顯示A陣列的元素內容：");  
for (int i = 0; i < A.length; i++)  
    System.out.print(A[i] + " ");
```

```
System.out.println("\n");
```

```
System.out.println("重新顯示B陣列的元素內容：");  
for (int i = 0; i < B.length; i++)  
    System.out.print(B[i] + " ");
```

```
}
```

```
}
```


複製陣列

```
1 public class CH04_07
2 {
3     public static void main(String[] args)
4     {
5         int[] A = new int[] { 2, 4, 6, 8, 10, 12 };
6         System.out.println("顯示A陣列的元素內容：");
7         for (int i = 0; i < A.length; i++)
8             System.out.print(A[i] + "\t");
9
10        System.out.println("\n\n複製A陣列的元素值到B陣列中...\n");
11
12        int[] B = new int[A.length];
13        for (int i = 0; i < B.length; i++)
14            B[i] = A[i];
15
16        System.out.println("顯示B陣列的元素內容：");
17        for (int i = 0; i < B.length; i++)
18            System.out.print(B[i] + "\t");
19
20        System.out.println("\n\n更改陣列A[1]的元素值中...\n");
21        A[1] = 2;
22
23        System.out.println("重新顯示A陣列的元素內容：");
24        for (int i = 0; i < A.length; i++)
25            System.out.print(A[i] + "\t");
26    }
```

複製陣列

```
27     System.out.println("\n");
28
29     System.out.println("重新顯示B陣列的元素內容：");
30     for (int i = 0; i < B.length; i++)
31         System.out.print(B[i] + "\t");
32     }
33 }
```

◆ 執行結果：

顯示A陣列的元素內容：

2 4 6 8 10 12

複製A陣列的元素值到B陣列中...

顯示B陣列的元素內容：

2 4 6 8 10 12

更改陣列A[1]的元素值中...

重新顯示A陣列的元素內容：

2 2 6 8 10 12

重新顯示B陣列的元素內容：

2 4 6 8 10 12

複製陣列

◆ 說明：

- 行05：
 - ◆ 宣告一維整數陣列A，並同時指定陣列的初值。
- 行06：
 - ◆ 螢幕輸出。
- 行07~行08：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `A.length=6`）。
- 行10：
 - ◆ 螢幕輸出。
- 行12：
 - ◆ 宣告一維整數陣列B，並指定陣列的大小與A陣列相同。

複製陣列

- 行13~行14：
 - ◆ 利用for迴圈，將A陣列中各個元素的值複製到B陣列相對應的位置。
- 行16：
 - ◆ 螢幕輸出。
- 行17~行18：
 - ◆ 利用for迴圈，輸出B陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `B.length=6`）。
- 行20：
 - ◆ 螢幕輸出。
- 行21：
 - ◆ 重新指定A[1]的元素值。
- 行23：
 - ◆ 螢幕輸出。

複製陣列

- 行24~行25：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `A.length=6`）。
- 行27：
 - ◆ 螢幕輸出。
- 行29：
 - ◆ 螢幕輸出。
- 行30~行31：
 - ◆ 利用for迴圈，輸出B陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `B.length=6`）。

複製陣列

2、clone複製方式：

- ◆ 陣列是屬於物件的一種，所以也可以使用物件類別裡定義的clone()方法來複製陣列。
- ◆ 語法：

目標陣列名稱=(資料型態[])來源陣列名稱.clone();

例如：

```
arr2=(int[])arr1.clone();
```

複製陣列

◆ 程式：

```
public class CH04_08
{
    public static void main(String[] args)
    {
        int A[] = { 2, 4, 6, 8, 10, 12 };
        System.out.println("顯示A陣列的元素內容：");
        for (int i = 0; i < A.length; i++)
            System.out.print(A[i] + "t");

        System.out.println("\n\n複製A陣列的元素值到B陣列中...\n");
        int B[] = new int[A.length];
        B = (int[]) A.clone();
    }
}
```

複製陣列

```
System.out.println("顯示B陣列的元素內容：");
```

```
for (int i = 0; i < B.length; i++)
```

```
    System.out.print(B[i] + " ");
```

```
System.out.println();
```

```
}
```

```
}
```


複製陣列

```
1 public class CH04_08
2 {
3     public static void main(String[] args)
4     {
5         int A[] = { 2, 4, 6, 8, 10, 12 };
6         System.out.println("顯示A陣列的元素內容：");
7         for (int i = 0; i < A.length; i++)
8             System.out.print(A[i] + "\t");
9
10        System.out.println("\n\n複製A陣列的元素值到B陣列中...\n");
11        int B[] = new int[A.length];
12        B = (int[]) A.clone();
13
14        System.out.println("顯示B陣列的元素內容：");
15        for (int i = 0; i < B.length; i++)
16            System.out.print(B[i] + "\t");
17
18        System.out.println();
19    }
20 }
```

複製陣列

◆ 執行結果：

顯示A陣列的元素內容：

2 4 6 8 10 12

複製A陣列的元素值到B陣列中…

顯示B陣列的元素內容：

2 4 6 8 10 12

◆ 說明：

- 行05：
 - ◆ 宣告一維整數陣列A，並同時指定陣列的初值。
- 行06：
 - ◆ 螢幕輸出。
- 行07~行08：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `A.length=6`）。

複製陣列

- 行10：
 - ◆ 螢幕輸出。
- 行11：
 - ◆ 宣告一維整數陣列B，並指定陣列的大小與A陣列相同。
- 行12：
 - ◆ 將A陣列中各個元素的值複製（clone）到B陣列相對應的位置。
- 行14：
 - ◆ 螢幕輸出。
- 行15~行16：
 - ◆ 利用for迴圈，輸出B陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `B.length=6`）。
- 行18：
 - ◆ 螢幕輸出。

複製陣列

3、arraycopy複製方式：

- ◆ 陣列的另一種複製的方式，是使用System類別裡的arraycopy方法。

- ◆ 語法：

System.arraycopy(來源陣列,起始索引,目標陣列,存放位置,資料長度);

- ◆ 說明：

- 來源陣列：
 - ◆ 要被複製的陣列。
- 起始索引：
 - ◆ 整數型態，設定來源陣列要被複製的起始位置。
- 目標陣列：
 - ◆ 複製到的陣列。

複製陣列

- 存放位置：
 - ◆ 整數型態，複製到目標陣列第n個位置。
- 資料長度：
 - ◆ 複製到目標陣列的資料長度。

例如：

```
System.arraycopy(str1,0,str2,1,2);
```

複製陣列

◆ 程式：

```
public class CH04_09
{
    public static void main(String[] args)
    {
        int[] A = new int[] { 1, 2, 3, 4, 5 };
        int[] B = new int[A.length];

        System.out.println("顯示A陣列的元素內容：");
        for (int i = 0; i < A.length; i++)
            System.out.print(A[i] + " ");

        System.out.println("\n\n顯示B陣列的元素內容：");
        for (int i = 0; i < B.length; i++)
            System.out.print(B[i] + " ");
```

複製陣列

```
System.out.println("\n\n用arraycopy方法，複製A陣列的元素值到B陣  
列中...\n");
```

```
System.arraycopy(A, 0, B, 0, 5);
```

```
System.out.println("顯示B陣列的元素內容：");
```

```
for (int i = 0; i < B.length; i++)
```

```
    System.out.print(B[i] + " ");
```

```
}
```

```
}
```

複製陣列

```
1 public class CH04_09
2 {
3     public static void main(String[] args)
4     {
5         int[] A = new int[] { 1, 2, 3, 4, 5 };
6         int[] B = new int[A.length];
7
8         System.out.println("顯示A陣列的元素內容：");
9         for (int i = 0; i < A.length; i++)
10             System.out.print(A[i] + "\t");
11
12         System.out.println("\n\n顯示B陣列的元素內容：");
13         for (int i = 0; i < B.length; i++)
14             System.out.print(B[i] + "\t");
15
16         System.out.println("\n\n用arraycopy方法，複製A陣列的元素值到B陣列中...\n");
17         System.arraycopy(A, 0, B, 0, 5);
18
19         System.out.println("顯示B陣列的元素內容：");
20         for (int i = 0; i < B.length; i++)
21             System.out.print(B[i] + "\t");
22     }
23 }
```


複製陣列

◆ 執行結果：

顯示A陣列的元素內容：

1 2 3 4 5

顯示B陣列的元素內容：

0 0 0 0 0

用arraycopy方法，複製A陣列的元素值到B陣列中...

顯示B陣列的元素內容：

1 2 3 4 5

◆ 說明：

- 行05：

- ◆ 宣告一維整數陣列A，並同時指定陣列的初值。

- 行06：

- ◆ 宣告一維整數陣列B，並指定陣列的大小與A陣列相同。

複製陣列

- 行08：
 - ◆ 螢幕輸出。
- 行09~行10：
 - ◆ 利用for迴圈，輸出A陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `A.length=6`）。
- 行12：
 - ◆ 螢幕輸出。
- 行13~行14：
 - ◆ 利用for迴圈，輸出B陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `B.length=6`）。
- 行16：
 - ◆ 螢幕輸出。

複製陣列

- 行17：
 - ◆ 利用System的arraycopy的方法，複製A陣列的內容，到B陣列中。
- 行19：
 - ◆ 螢幕輸出。
- 行20~行21：
 - ◆ 利用for迴圈，輸出B陣列中各個元素的值。
 - ◆ 陣列的長度為6（即 `B.length=6`）。

複製陣列

■ 複製陣列的三種方法中，

- 使用clone的方法，速度最慢，尤其是陣列愈大愈明顯，但程式碼簡單，且不論是一維陣列或是多維陣列，均使用相同的程式碼，最為方便。
- 使用System的arraycopy方法，速度最快，且可以複製指定的範圍（包括來源範圍與目的位置等）。
- 用for迴圈，會比arraycopy方法更有彈性。

■ 所以三種方法，各有優、缺點，要使用哪種方式複製陣列，端看程式的實際需求。

資料來源

- 蔡文龍、何嘉益、張志成、張力元，JAVA SE 10基礎必修課，台北市，碁峰資訊股份有限公司，2018年7月，出版。
- 吳燦銘、胡昭民，圖解資料結構-使用Java(第三版)，新北市，博碩文化股份有限公司，2018年5月，出版。
- Ivor Horton，Java 8 教學手冊，台北市，碁峰資訊股份有限公司，2016年9月，出版。
- 李春雄，程式邏輯訓練入門與運用---使用JAVA SE 8，台北市，上奇科技股份有限公司，2016年6月，初版。
- 位元文化，Java 8視窗程式設計，台北市，松崗資產管理股份有限公司，2015年12月，出版。
- Benjamin J Evans、David Flanagan，Java 技術手冊 第六版，台北市，碁峰資訊股份有限公司，2015年7月，出版。
- 蔡文龍、張志成，JAVA SE 8 基礎必修課，台北市，碁峰資訊股份有限公司，2014年11月，出版。
- 陳德來，Java SE 8程式設計實例，台北市，上奇科技股份有限公司，2014年11月，初版。
- 林信良，Java SE 8 技術手冊，台北市，碁峰資訊股份有限公司，2014年6月，出版。
- 何嘉益、黃世陽、李篤易、張世杰、黃鳳梅，徐政棠譯，JAVA2 程式設計從零開始--適用JDK7，台北市，上奇資訊股份有限公司，2012年5月，出版。