第四章　陣列

備註：可依進度點選小節

4.1 為何需要陣列

4.1.1 何謂陣列(Array)

- 前面章節，每使用到一個資料就需宣告一個變數來存放，資料一多時，變數亦跟著增加，增加變數命名困擾且程式長度亦增長不易維護。
- C# 對相同性質的資料提供陣列來存放。
- 在宣告陣列時
  ① 設定陣列名稱
  ② 建立陣列大小
  ③ 陣列的資料型別

C# 在編譯時自動在記憶體中保留連續空間來存放該陣列所有元素。

陣列的宣告與建立

- 方式1：

  先宣告陣列名稱，再使用new關鍵字建立陣列的大小

  資料型別 [] 陣列名稱;

  陣列名稱 = new 資料型別 [大小];

- 方式2：

  宣告陣列的同時並使用 new 關鍵字建立陣列的大小

  資料型別 [] 陣列名稱 = new 資料型別 [大小];

- 如： int[] myAry = new int[5] ;

- 陣列名稱後面中括號內的整數值稱為 註標 或 索引。
- 若將註標以變數取代，在程式中欲存取陣列元素只要改變註標值即可。
- 可將一個陣列元素視為一個變數，也就是將 myAry[0] ~ myAry[4] 視為5個變數名稱
- 變數間以註標區別，可免變數命名困擾。
- 由於陣列經過宣告，在編譯時期會保留連續記憶體位址給該陣列中的元素使用
- 陣列元素會依註標先後次序存放在這連續的記憶體位址存取陣列元素只要指定陣列的註標， C# 透過註標自動計算出該陣列元素的位址來存取指定的陣列元素。

- **初始化 (Initialization)**

  陣列建立完畢，便可透過下列指定陳述式(=)直接在程式中設定各陣列元素的初值。

```
myAry[0] = 10;
myAry[1] = 20;
myAry[2] = 30;
myAry[3] = 40;
myAry[4] = 50;
```

- 上面陳述式是將陣列的建立和初值分開書寫，若希望在建立同時就設定陣列的初值。
- 語法：

  資料型別 [] 陣列名稱 = new 資料型別 [大小] {陣列初值};

- 將上面 myAry 陣列的建立和初值設定共六行陳述式合併成一行，其寫法如下：

```
int[] myAry = new int[5] {10,20,30,40,50};
```

或

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

- 若建立陣列時未設定初值
  ① 若是數值資料型別預設值為零
  ② 若是字串資料型別預設為null
  ③ 布林資料型別預設為false。

- 假設陣列元素由記憶位址1,000 開始放起且每個記憶體位址大小只允許存放1 Byte的資料
- 一個整數變數用4 Bytes存放資料，需佔用四個記憶體位址。
- 經過建立和設定初值後，各陣列元素的記憶位址和內容如下：

陣列元素	內容	實際配置記憶位址
myAry[0]	10	1000~1003
myAry[1]	20	1004~1007
myAry[2]	30	1008~1011
myAry[3]	40	1012~1015
myAry[4]	50	1016~1019

4.1.2 一維陣列的存取

- 同性質資料用陣列來存放，可透過 for迴圈配合變數 k 當陣列註標，逐一將鍵入資料存入陣列，也可將資料由陣列中讀取出來。
- 下例以 變數k 當計數，並將對應的 k 值當做陣列元素的註標，連續由鍵盤讀取資料五次，便可放入陣列a：

Step1 當k=0，透過Console.ReadLine() 方法將輸入值置入a[k] 即a[0]。

Step2 當k=1，透過Console.ReadLine() 方法將輸入值置入a[k] 即a[1]。

Step3 當k=2，透過Console.ReadLine() 方法將輸入值置入a[k] 即a[2]。

Step4 當k=3，透過Console.ReadLine() 方法將輸入值置入a[k] 即a[3]。

Step5 當k=4，透過Console.ReadLine() 方法將輸入值置入a[k] 即a[4]。

- 將上面步驟寫成程式片段：

```
for (k=0 ;k<=4 ; k++) {
    a[k] = int.Parse (Console.ReadLine());
     或
    a[k] =  Convert.ToInt32(Console.ReadLine());
 }
```

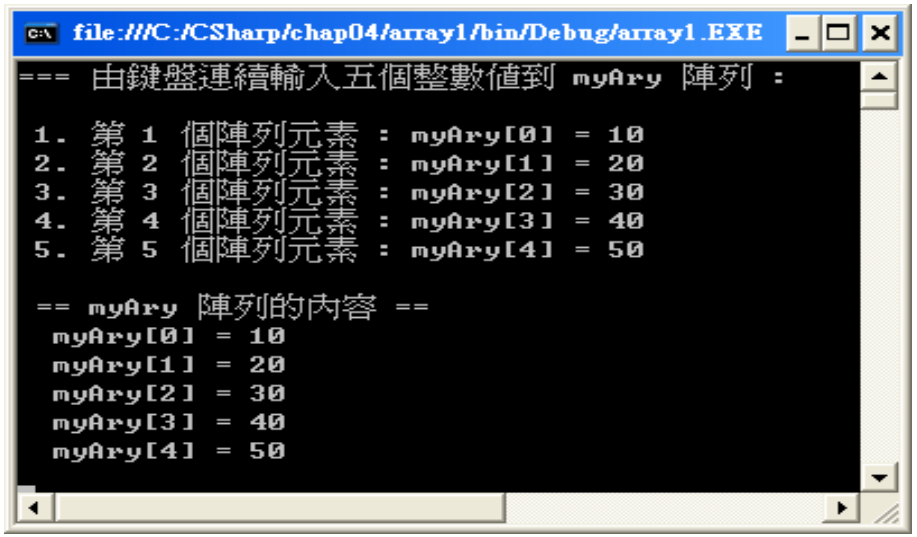
- 使用 for 迴圈讀取陣列a 中所有陣列元素的內容，寫法：

```
for (k=0 ;k<=4 ; k++) {
    Console.WriteLine ("{0} ", a[k]);
 }
```

範例演練

請參照下圖的輸出入畫面，將本小節所介紹使用 for 迴圈來存取陣列的內容寫成一個完整程式。

執行時，先由鍵盤連續輸入五個整數存到 myAry[0] ~ myAry[4] 陣列元素內，最後再將 myAry[0] ~ myAry[4] 陣列元素內容印出來。



```
// FileName: array1.sln
01  static void Main(string[] args)
02  {
03     int k;
04     int[] myAry = new int[5];
05     Console.WriteLine ("=== 由鍵盤連續輸入五個整數值到 myAry 陣列 : \n ");
06     for (k = 0; k < 5; k++)
07     {
08        Console.Write(" {0}. 第 {1} 個陣列元素 : myAry[{2}] = ", k + 1, k + 1, k);
10        myAry[k] = int.Parse(Console.ReadLine());
11     }
12     Console.WriteLine();   // 空一行
13     Console.WriteLine(" == myAry 陣列的內容 == ");
14     for (k = 0; k < 5; k++)   // 顯示myAry[0]~myAry[4]
15     {
16        Console.WriteLine("  myAry[{0}] = {1}", k, myAry[k]);
17     }
18     Console.Read();
19  }
```

4.2 陣列常用的屬性與方法

- 陣列物件被建立時(實體化)，即可用陣列物件提供的方法與屬性。
- 透過方法可取得陣列的相關資訊，如：陣列的維度 陣列元素個數...等。
- 一維陣列

```
int[] ary1 = new int[] { 1, 2, 3, 4, 5 };
```

- 二維陣列

```
int[,] ary2 = new int[,] { {1,2,3 }, {4,5,6 }, {7,8,9 }, {10,11,12 } };
```

陣列物件的成員	說明

Length 屬性	取得陣列元素的總數。
	[例] int a1=ary1.Length ; //a1=5, ary1 陣列元素總數
	int a2=ary2.Length; //a2=12, ary2 陣列元素總數

Rank 屬性	取得陣列維度數目。
	[例] int r1=ary1.Rank ; // r1=1
	int r2=ary2.Rank; // r2=2

GetUpperBound 方法	取得陣列某一維度的上限。
	[例]
	int U1=ary1.GetUpperBound(0); //U1=4,第 1 維上限
	int U2=ary2.GetUpperBound(1); //U2=2,第 2 維上限

GetLowerBound 方法	取得陣列某維度的下限，陣列維度下限由 0 開始。

GetLength 方法	取得陣列某一維度的陣列元素總數。
	[例] int t1=ary1.GetLength(0); //t1=5,第 1 維元素總數
	int t2=ary2.GetLength(1); //t2=3,第 2 維元素總數

4.3 Array 類別常用靜態方法

- Array 類別即陣列類別是支援陣列實作的基底類別，用來提供建立、管理、搜尋和排序陣列物件的方法。
- 類別靜態方法
  就是指類別不用實體化為物件可直接呼叫該靜態方法
- 下面介紹 Array 類別靜態方法僅限用在一維陣列的處理上。

4.3.1 陣列的排序

- **Array.Sort() 方法**

  可用來對指定的一維陣列物件由小而大做遞增排序。

- **語法1：將一維陣列物件中的元素做由小到大排序**

  Array.Sort(陣列物件);

- **語法2：**

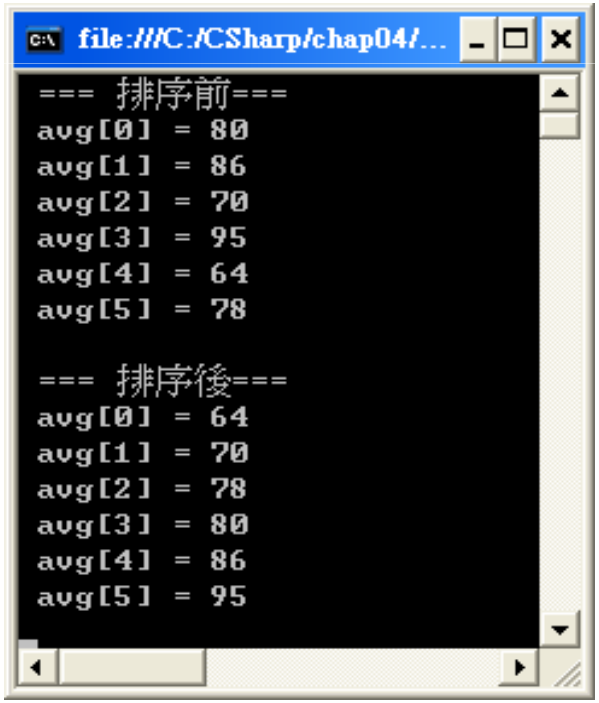
  用來將 陣列物件1 中的元素做由小到大排序，
  且 陣列物件2 的元素會隨著 陣列物件1 的
  索引位置跟著做排序的動作。

  Array.Sort(陣列物件1, 陣列物件2);

。

範例演練

先在程式中直接設定陣列元素的初值，再透過 for 迴圈顯示陣列的初值。接著用 Array.Sort() 方法做遞增排序再顯示排序後的結果。



FileName: ArraySort1.sln

```
01  static void Main(string[] args)
02  {
03     int[] avg = new int[6] { 80, 86, 70, 95, 64, 78 };
04     Console.WriteLine(" === 排序前=== ");
05     for (int k=0; k<=avg.GetUpperBound(0); k++)
06     {
07        Console.WriteLine(" avg[{0}] = {1}", k, avg[k]);
08     }
09     Console.WriteLine();   //換行
10     Array.Sort(avg);
11     Console.WriteLine(" === 排序後=== ");
12     for (int k=0; k<=avg.GetUpperBound(0); k++)
13     {
14        Console.WriteLine(" avg[{0}] = {1}", k, avg[k]);
15     }
16     Console.Read();
17  }
```

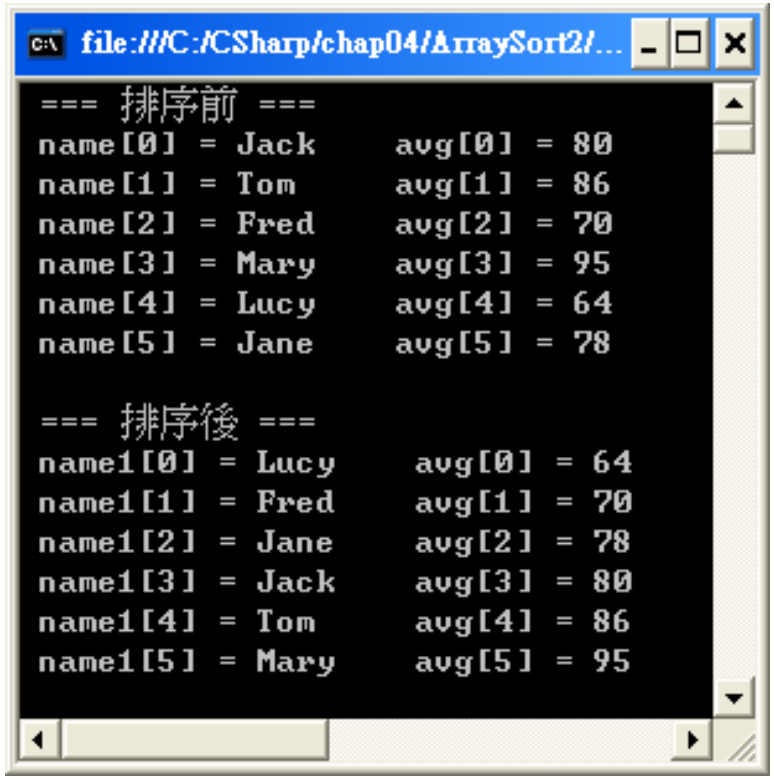
範例演練

下表為某個班級的學期成績，由於此表中有兩個不同性質的資料，因此必須使用兩個陣列來分別存放姓名和學期成績。假設陣列名稱分別為 name 和 avg 的初值如左下圖所示：

姓名(name)	學期成績(avg)
Jack	80
Tom	86
Fred	70
Mary	95
Lucy	64
Jane	78

姓名(name)	學期成績(avg)
Lucy	64
Fred	70
Jane	78
Jack	80
Tom	86
Mary	95

- 排序時用 Array.Sort(avg)，只單獨對avg陣列物件做遞增排序時，name姓名陣列仍維持原狀，導致姓名和成績無法一致。
- 改成Array.Sort(avg, name)，學期成績排序結果如圖所示：



```
// FileName: ArraySort2.sln
01  static void Main(string[] args)
02  {
04      string[] name = new string[6] { "Jack", "Tom ", "Fred", "Mary", "Lucy", "Jane" };
06      int[] avg = new int[6] { 80, 86, 70, 95, 64, 78 };
07      Console.WriteLine(" === 排序前 === ");
08      for (int k = 0; k <= avg.GetUpperBound (0); k++)
09      {
10          Console.WriteLine(" name[{0}] = {1}    avg[{2}] = {3}", k, name[k] , k, avg[k] );
11      }
12      Console.WriteLine();
13      Array.Sort(avg, name);
14      Console.WriteLine(" === 排序後 === ");
15      for (int k = 0; k <= avg.GetUpperBound (0); k++)
16      {
17          Console.WriteLine(" name1[{0}] = {1}   avg[{2}] = {3}", k, name[k] , k, avg[k] );
18      }
19      Console.Read();
20  }
```

4.3.2 陣列的反轉

- **Array.Reverse() 方法**

  用來反轉整個一維陣列的順序。

- 上例用Array.Sort() 方法對指定陣列由小而大遞增排序。

- 若希望改由大而小作遞減排序，需再將已做完遞增排序的陣列再用 Array.Reverse() 方法即可將陣列由大而小作遞減排序。

- Array.Reverse() 語法：

  Array.Reverse(陣列物件);

- [例] 欲對陣列名稱avg做由大而小遞減排序，寫法：

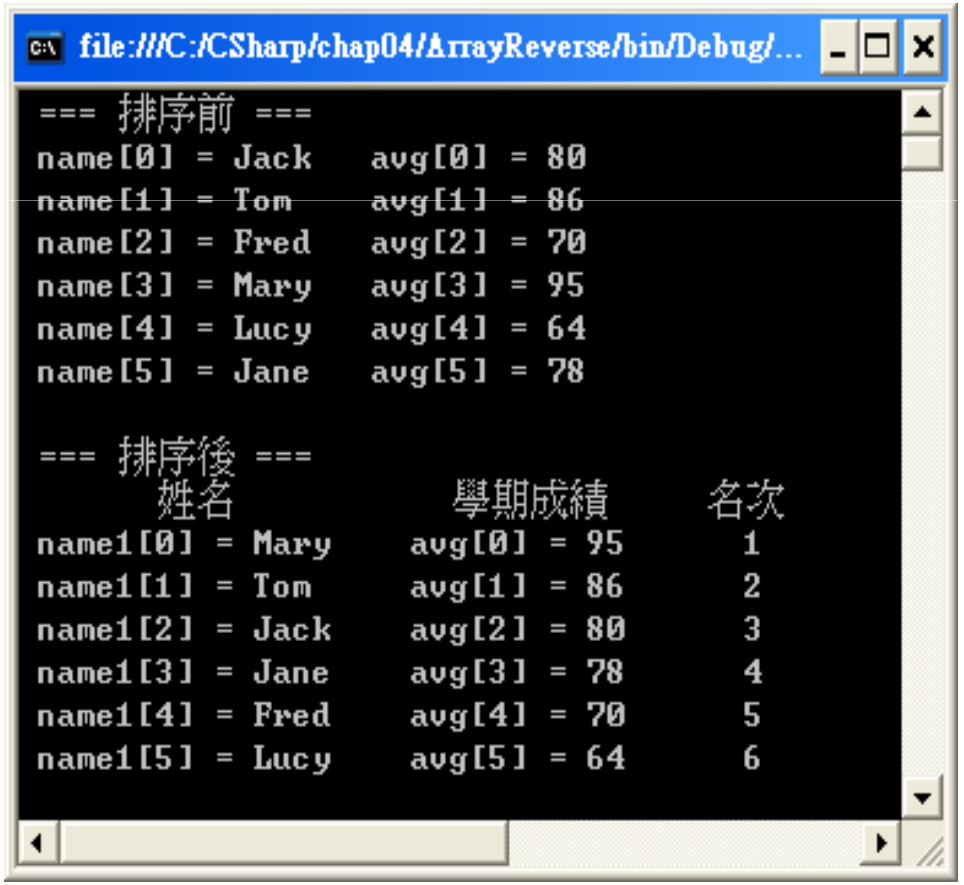
```
Array.Sort(avg);
Array.Reverse(avg);
```

- 若同時有兩個相關陣列 name 和 avg，若以 avg 陣列為基準由大而小做遞減排序，相關陣列需要同時反轉，寫法：

```
Array.Sort(avg,name);
Array.Reverse(avg);
Array.Reverse(name);
```

範例演練

延續上例，除成績採遞減排序且在輸出結果前加上姓名、學期成績及名次。



```
// FileName: ArrayReverse.sln
01  static void Main(string[] args)
02  {
03      string[] name = new String[6] { "Jack", "Tom ", "Fred", "Mary",
                          "Lucy", "Jane" };
04      int[] avg = new int[6] { 80, 86, 70, 95, 64, 78 };
05      Console.WriteLine(" === 排序前 === ");
06      for (int k = 0; k <= avg.GetUpperBound (0); k++)
07       {
08          Console.WriteLine(" name[{0}] = {1}   avg[{2}] = {3}",
                                  k, name[k], k, avg[k]);
10       }
```

```
11    Console.WriteLine();
12    Array.Sort(avg, name);
13    Array.Reverse(avg);
14    Array.Reverse(name);
15    Console.WriteLine(" === 排序後 === ");
16    Console.WriteLine("      姓名          學期成績    名次 ");
17    for (int k = 0; k <= avg.GetUpperBound (0); k++)
18    {
19        Console.WriteLine(" name1[{0}] = {1}  avg[{2}] = {3}  {4}",
                                k, name[k], k, avg[k], k + 1);
20    }
21    Console.Read();
22  }
```

4.3.3 陣列的搜尋

- .NET Framework 類別程式庫的 Array 類別提供
  1. Array.IndexOf()方法
  2. Array. BinarySearch() 方法

  用來搜尋某個資料是否在陣列物件中。

1. Array.IndexOf()方法

使用 Array.IndexOf 可用來搜尋陣列中是否有相符資料。

① 若有找到，則會傳回該陣列元素的註標值。

② 若沒有找到，會傳回-1。

語法：

```
Array.IndexOf(陣列名稱,查詢資料[,起始註標][,查詢距離]);
```

[例] 假設字串陣列 name 中有
{"Jack","Tom","Fred","Mary","Lucy", "Jane" }
共六個陣列元素，觀察下列各陳述式輸出結果：

① Array.IndexOf(name,"Tom");

[結果] 由註標0開始找起，傳回值為1。

② Array.IndexOf(name, "Tom", 3) ;

[結果] 由註標3開始找起，傳回值為-1。

③ 若str1="Lucy" , start=1, offset=2
Array.IndexOf(name, str1, start, offset);

[結果] 由註標1開始往下找2個陣列元素的內容是否有 " Lucy" 字串。傳回值為-1。

2. Array.BinarySearch()方法

- 用來搜尋陣列中的資料，陣列未經排序，每次搜尋資料都由最前面開始，資料量大時，愈後面的資料查詢所花費的時間愈多，資料平均搜尋時間不平均。
- 為不管資料前後次序，使得資料平均搜尋時間都差不多，在 .NET Framework 類別程式庫另提供此二分化搜尋方法來搜尋資料是否在陣列中。
- 此方法使用前陣列需先經由小而大排序才可使用適用於資料量大的陣列。

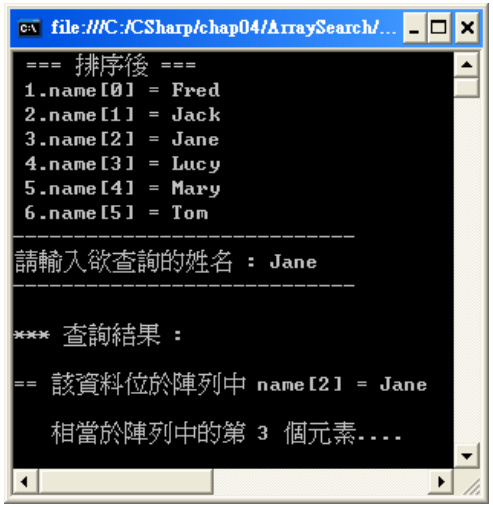
語法：Array.BinarySearch(陣列名稱, 查詢資料);

範例演練

延續上例，取 name 姓名陣列，先將陣列由小而大做遞增排序，接著由鍵盤輸入欲查詢的英文名字，透過 Array.BinarySearch()方法查詢：

① 若有找到即顯示該陣列元素的註標和內容以及顯示是第幾個陣列元素；

② 若找不到該資料，則顯示 “該資料不存在!“。



```
// FileName: ArraySearch.sln
01   static void Main(string[] args)
02   {
03      int index;
04      string myobject;
05      string[] name = new string[6] { "Jack", "Tom", "Fred", "Mary", "Lucy", "Jane" };
06      Array.Sort(name);.
07      Console.WriteLine(" === 排序後 === ");
08      for (int k = 0; k <= 5; k++)
09      {
10         Console.WriteLine(" {0}.name[{1}] = {2} ", k + 1, k, name[k]);
11      }
```

```
12    Console.WriteLine("--------------------------");
13    Console.Write("請輸入欲查詢的姓名 : ");
14    myobject = Console.ReadLine();
15
16    index = Array.BinarySearch(name, myobject);
17    Console.WriteLine("--------------------------");
18    Console.WriteLine();
19    Console.WriteLine("*** 查詢結果 : ");
20    Console.WriteLine();
21    if (index < 0)   //index小於0，表示找不到資料
22    {
23      Console.WriteLine("== 該資料不存在 !");
24    }
25  Else
26     {
27       Console.WriteLine("== 該資料位於陣列中 name[{0}] = {1}",
                                index, name[index]);
28       Console.WriteLine("\n   相當於陣列中的第 {0} 個元素....", index + 1);
29     }
30     Console.Read();
31  }
```

4.3.4 陣列的拷貝

- 將某個陣列複製給另一個陣列時，可用 Array.Copy() 方法進行拷貝陣列。
- 語法：

```
Array.Copy (srcAry , srcIndex , dstAry , dstIndex , length );
```

①srcAry　：來源陣列即被拷貝的陣列。

②srcIndex：代表 ***srcAry*** 來源陣列的註標，由指定的註標開始複製。

③dstAry　：接收資料的目的陣列。

④dstIndex：代表 ***dstAry*** 目的陣列的註標，由指定的註標開始儲存。

⑤length　：表示要複製的陣列元素個數。

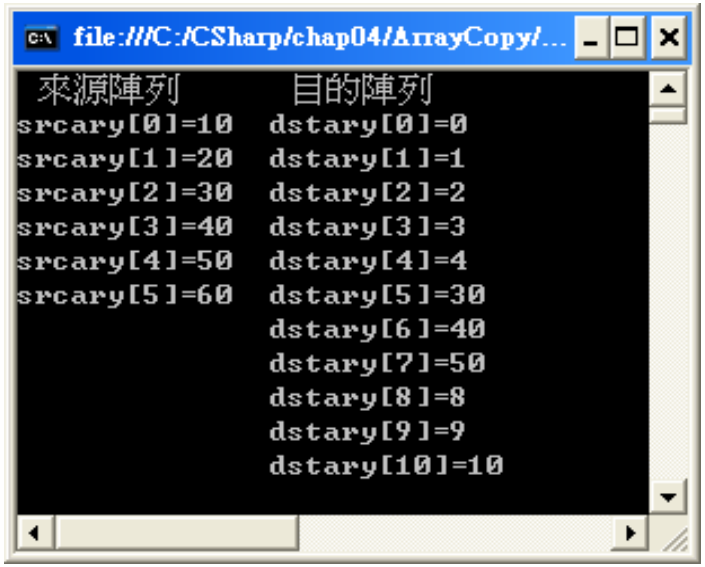
範例演練

練習使用 Array.Copy()方法來進行拷貝陣列。先建立

①來源陣列：int[] srcary=new int[] {10, 20, 30, 40, 50, 60};

②目的陣列：int[] dstary=new int[] {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

將 srcary 來源陣列註標值為 2 開始往下拷貝 3 個陣列元素到 dstary 目的陣列，並從 dstary目的陣列的第5個註標開始放起。



```
// FileName: ArrayCopy.sln
01  static void Main(string[] args)
02  {
03      //建立來源陣列
04      int[] srcary = new int[] { 10, 20, 30, 40, 50, 60 };
05      //建立目的陣列
06      int[] dstary = new int[] { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
07      Array.Copy(srcary, 2, dstary, 5, 3);
08      Console.WriteLine(" 來源陣列      目的陣列");
09      for (int k = 0; k <= 10; k++) {
11          if (k <= 5)  {
13              Console.WriteLine("srcary[{0}]={1}  dstary[{2}]={3}", k,
                                        srcary[k], k, dstary[k]);
14          }
15          else {
17              Console.WriteLine("          dstary[{0}]={1}", k, dstary[k]);
18          }
19      }
20      Console.Read();
21  }
```

4.3.5 陣列的清除

- 當需要將某個陣列中指定範圍內的陣列元素的內容清除，可透過 Array.Clear() 方法。語法：

  Array.Clear(aryname, startindex, length);

【例1】將 myary 陣列中，註標為 3~4 陣列元素的內容清除， 寫法：

```
Array.Clear(myary, 3, 2);
```

【例2】 將 myary 陣列中所有陣列元素的內容清除，假設該 陣列共有六個陣列元素。寫法：

```
Array.Clear(myary, 0, 6);
```

4.4 多維陣列

- **一維陣列** (One-Dimensional Array)
  陣列只有一個註標，維度為1者。
- **二維陣列** (Two-Dimensional Array)
  陣列有兩個註標，維度為2者。
- **三維陣列** (Three-Dimensional Array)
  陣列有三個註標，維度為3者。
- **多維陣列**(Multi-Dimensional Array)
  維度超過兩個(含)以上者。

二維陣列

- 是由兩個註標構成。
- 將第一個註標稱為列(Row)，第二個註標稱為行(Column)。
- 如座位表、電影座位等以表格方式呈現者都可二維陣列來表示。
- 二維陣列若每列的個數都相同，構成下頁矩形陣列。
- 若每列的個數長短不一，就構成不規則陣列(Jagged Array)。

	第 0 行	第 1 行	第 2 行	第 3 行
第 0 列	ary2[0,0]	ary2 [0,1]	ary2[0,2]	ary2[0,3]
第 1 列	ary2 [1,0]	ary2[1,1]	ary2[1,2]	ary2[1,3]
第 2 列	ary2 [2,0]	ary2[2,1]	ary2[2,2]	ary2[2,3]

陣列名稱　列註標　行註標

- 上表二維陣列建立方式：

  int[] ary2 = new int[3,4];

- 設定各陣列元素的初值：

  ary2[0,0]=1 ; ary2[0,1]=2 ; ary2[0,2]=3 ; ary2[0,3]=4;

  ary2[1,0]=5 ; ary2[1,1]=6 ; ary2[1,2]=7 ; ary2[1,3]=8;

  ary2[2,0]=9 ; ary2[2,1]=10 ; ary2[2,2]=11 ; ary2[2,3]=12;

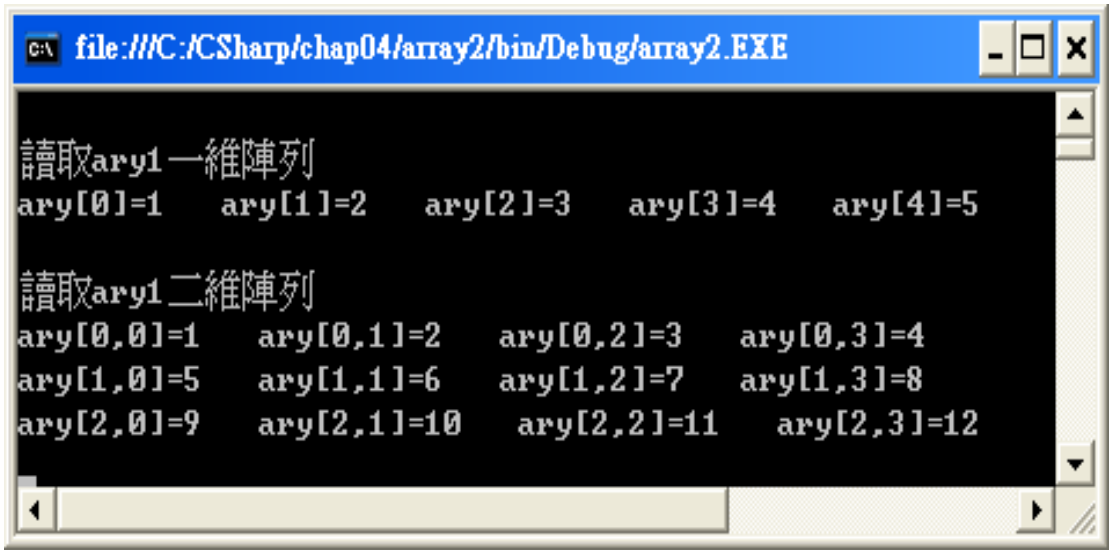
- 將上面建立和設定初值陳述式合併成一行敘述：

  int [,]ary2 = new int[,] {{1,2,3,4}, {5,6,7,8},{9,10,11,12}};

範例演練

參照下圖輸出入畫面，使用陣列物件所提供的方法配合 for 迴圈，將下列的 ary1 一維陣列及 ary2 二維陣列的所有元素讀取出來。

```
int[] ary1 = new int[] { 1, 2, 3, 4, 5 };
int[,] ary2 = new int[,] { { 1, 2, 3, 4 }, { 5, 6, 7, 8 }, { 9, 10, 11, 12 } };
```



```
// FileName: array2.sln
01   static void Main(string[] args)
02   {
04       int[] ary1 = new int[] { 1, 2, 3, 4, 5 };
05       int[,] ary2 = new int[,] { { 1, 2, 3, 4 }, { 5, 6, 7, 8 }, { 9, 10, 11, 12 } };
06       Console.WriteLine();
07       Console.WriteLine("讀取ary1一維陣列");
08       // 如下for可改成 for (int i = 0; i < ary1.Length ; i++)
09       for (int i = 0; i <= ary1.GetUpperBound(0); i++)
10       {
11          Console.Write("ary[{0}]={1}   ", i, ary1[i]);
12       }
```

```
13      Console.WriteLine();  //換行
14      Console.WriteLine();  //換行
15      Console.WriteLine("讀取ary1二維陣列");
16      //外層迴圈取得第1維陣列上限
17      for (int i = 0; i <= ary2.GetUpperBound(ary2.Rank - 2); i++)
18      {
19        //內層迴圈取得第2維陣列上限
20        for (int j = 0; j <= ary2.GetUpperBound(ary2.Rank - 1); j++)
21        {
22          Console.Write("ary[{0},{1}]={2}   ", i, j, ary2[i, j]);
23        }
24        Console.WriteLine();
25      }
26      Console.Read();
27  }
```

範例演練

由鍵盤如下圖由上而下輸入各選區每位候選人的得票數以行為主Column-Majored 方式存入陣列，輸入完畢電腦自動計算每位候選人的總得票數，以下表方式顯示，並顯示哪位候選人當選及得票數訊息。

候選人	第一選區	第二選區	第3選區	總得票數
周傑輪	10000	50000	70000	130000
榮一林	20000	60000	90000	170000
羅字祥	30000	40000	80000	150000

FileName: election.sln

```
01   static void Main(string[] args)
02   {
03       int i, k;
04
05       string[] name = new string[] { "周傑輪", "菜一林", "羅字詳" };
06       int[] tot = new int[name.Length];
07       int[,] vote = new int[3, 3];
08
09       for (i = 0; i <= 2; i++)
10       {
11           Console.WriteLine(" 第 {0} 選區各明星得票數:", i + 1);
12           for (k = 0; k <= 2; k++)
13           {
```

```
14              Console.Write(" {0}. {1} :", (k + 1), name[k]);
15              vote[i, k] = int.Parse(Console.ReadLine());
16          }
17          Console.WriteLine(" --------------------------------");
18      }
19      // 計算各候選人總得票數存入tot陣列中
24      for (i = 0; i <= 2; i++)
25      {
26          for (k = 0; k <= 2; k++)
27          {
28              tot[i] += vote[k, i];
29          }
30      }
31      // 顯示結果
```

```
32      Console.WriteLine(" ==================================");
33      Console.WriteLine(" 候選人 第一區  第二區 第三區 總得票數");
34      Console.WriteLine(" ====== ======  ======  ======  =======");
35      for (i = 0; i <= 2; i++)
36      {
37          Console.WriteLine(" {0}  {1}   {2}   {3}   {4}",
                              name[i], vote[0, i], vote[1, i], vote[2, i], tot[i]);
38      }
39      // 對存放各候選人總得票數的tot陣列作遞減排序
40      Array.Sort(tot, name);
41      Array.Reverse(tot);
42      Array.Reverse(name);
43      Console.WriteLine();
44      Console.WriteLine(" === {0} 獲得最高票, 共計: {1} 票", name[0], tot[0]);
45      Console.Read();
46  }
```

4.5 不規則陣列 (Jagged Array)

- 不規則陣列
  即是陣列元素再指向一個一維陣列，和矩陣陣列不一樣地方在於每列長度(即陣列元素的個數)不相同。
- 使用時機
  當在程式中建立一個二維陣列，若每列陣列元素個數長短不一時或有少數列陣列元素個數很大，其它列的陣列元素個數很少時，就可用不規則陣列
- 可使陣列佔用較少記憶體空間，執行時陣列存取速度較快。

- **Step1** 宣告二維陣列，先建一維陣列元素大小。

  先宣告不規則整數二維陣列，但第一維陣列先建立myary[0]~myary[2] 的整數陣列元素。

  ```
  int [][] myary = new int[3][] ; //先建立第一維有3列
  ```

- **Step2**

  經上面建立一維陣列之後，接著再對一維陣列的每一個元素使用new關鍵字建立新的一維陣列，且新的一維陣列的大小都不一樣，如此即形成不規則陣列。

  寫法：

  ```
  myary[0]=new int[] {1,2};        // 第0列myary[0][0]~myary[0][1]
  myary[1]=new int[] {3,4,6,7};    // 第1列myary[1][0]~myary[1][3]
  myary[2]=new int[] {8};          // 第2列myary[2][0]
  ```

Step3

由於每列的個數不一樣，就必須透過陣列的 Length 屬性來取得該維度的最大值。譬如：

① 用來取得整個myary陣列共有多少列

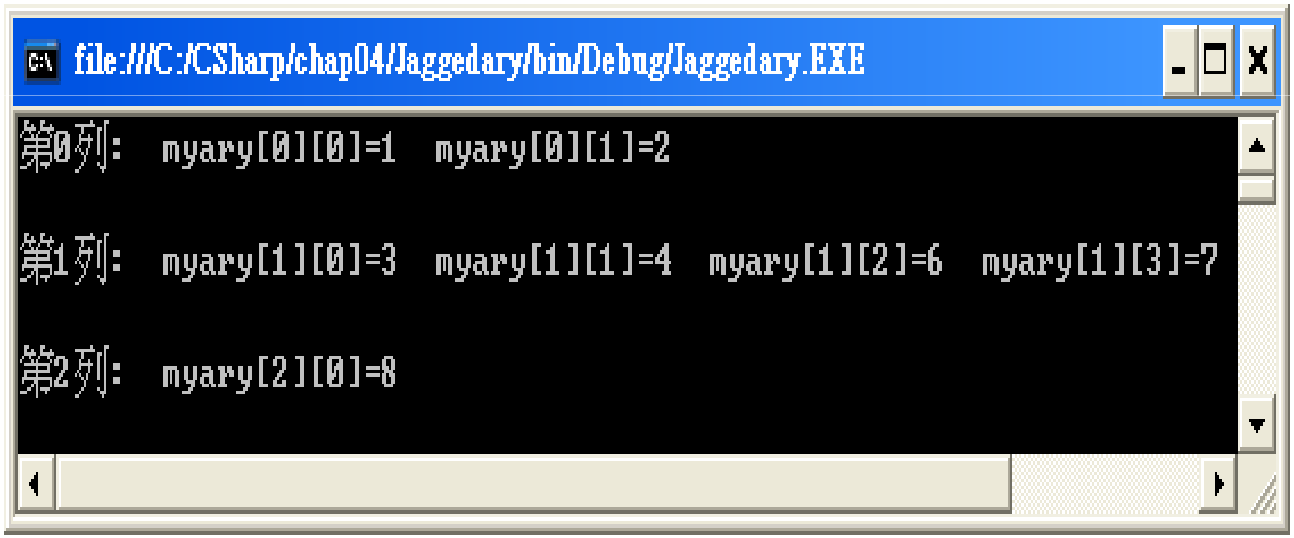
myary.Length

②用來取得myary陣列的第i列共有多少個陣列元素。

myary[i].Length

範例演練

將上面建立的不規則陣列中所有元素的內容，
按下圖結果全部顯示出來。



```
// FileName : Jaggedary.sln
01  static void Main(string[] args)
02  {
03      int[][] myary = new int[3][];
04      myary[0] = new int[] { 1, 2 };
05      myary[1] = new int[] { 3, 4, 6, 7 };
06      myary[2] = new int[] { 8 };
07      //取得整個myary陣列共有多少列
08      for (int i = 0; i < myary.Length; i++)
09      {
11          Console.Write("第{0}列: ", i);
13          for (int k = 0; k < myary[i].Length; k++)
14          {
15              Console.Write(" myary[{0}][{1}]={2} ", i, k, myary[i][k]);
16          }
17          Console.WriteLine("\n");
18      }
19      Console.Read();
20  }
```