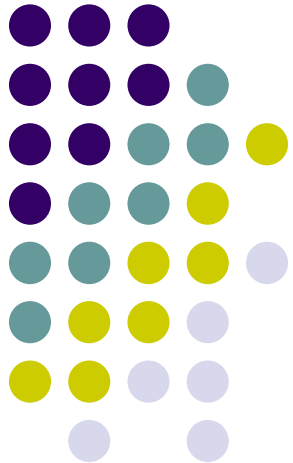
第八章
陣列與字串

認識一維與二維以上的陣列

瞭解陣列元素的表示方法

迴圈與陣列的使用



變數vs土地

```
int num1=12400;
```

宣告整數變數 num1，
並設值為 12400

- 變數宣告：由作業系統分配閒置的記憶體空間儲存資料

```
int num; //宣告
num = 12400; //初始化
```

  - 變數名稱不會重複
  - 變數位置: 0x10001000
  - 僅宣告未初始的變數?

- 土地使用：向地政事務所登記持有的門牌地址作為特定用途

```
學校 大學路1號; //登記
大學路1號 = 成大; //建設
```

  - 門牌地址不會重複
  - 門牌位置: (23.0, 120.2)
  - 僅登記未建設的地址?

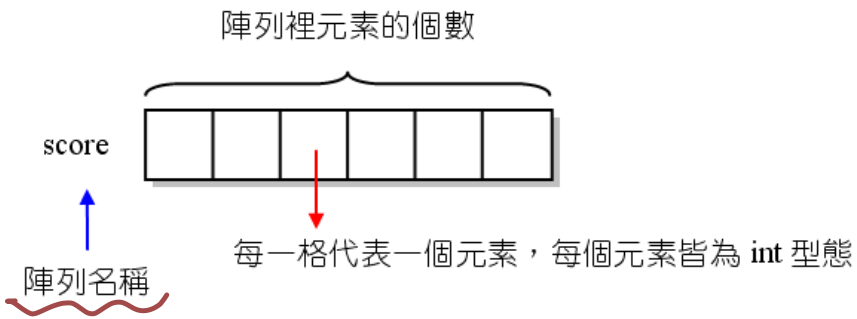
一維陣列的宣告

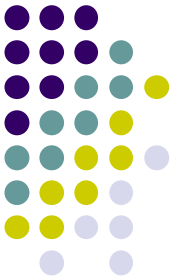
- 陣列可用來存放相同型態的元素
- 一維陣列的宣告格式如下所示

```
資料型態 陣列名稱[個數];     // 宣告一維陣列
```

- 下面的範例都是合法的一維陣列宣告

```
int score[6];
float temp[7];
char name[12];
```





陣列與元素的長度

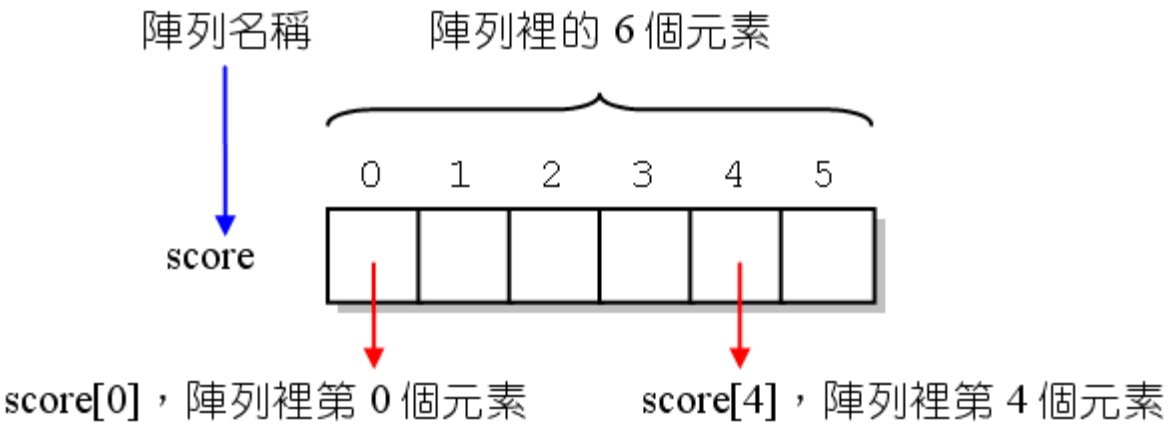
- 利用sizeof() 印出陣列score與其中任一個元素的長度

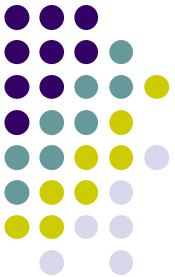
```
01  // prog8_1, 一維陣列
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int score[6];
08
09     // 印出陣列中個別元素的長度及陣列的總長度
10     cout << "sizeof(score[1])=" << sizeof(score[1]) << endl;
11     cout << "sizeof(score)=" << sizeof(score) << endl;
12     system("pause");
13     return 0;
14  }
```

```
/* prog8_1 OUTPUT------

sizeof(score[1])=4
sizeof(score)=24

----------------------*/
```

陣列元素的表示方法

- 陣列中的元素是以索引值來標示存放的位置
- 陣列索引值的編號必須由0開始
- 下圖為score陣列中元素的表示法及排列方式



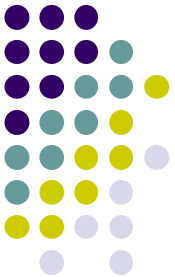


陣列基本操作

- 一維陣列的基本操作：

```
01  /* prog9_1, 一維陣列的基本操作 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int i,score[4];     /* 宣告整數變數 i 與整數陣列 score */
07
08     score[0]=78;        /* 設定陣列的第一個元素為 78 */
09     score[1]=55;        /* 設定陣列的第二個元素為 55 */
10     score[2]=92;        /* 設定陣列的第三個元素為 92 */
11     score[3]=80;        /* 設定陣列的最後一個元素為 80 */
12
13     for(i=0;i<=3;i++)
14        printf("score[%d]=%d\n",i,score[i]);  /* 印出陣列的內容 */
15
16     system("pause");
17     return 0;
18  }
```

```
/* prog9_1 OUTPUT---

score[0]=78
score[1]=55
score[2]=92
score[3]=80
--------------------*/
```



陣列元素的輸入

- 由鍵盤輸入資料來設定陣列元素：

```
01  /* prog9_4, 一維陣列內元素的設值 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int i,age[3];
07     for(i=0;i<3;i++)
08     {
09        printf("請輸入 age[%d]的值:",i);
10        scanf("%d",&age[i]);  /* 由鍵盤輸入數值給陣列 age 裡的元素 */
11     }
12     for(i=0;i<3;i++)
13        printf("age[%d]=%d\n",i,age[i]);
14
15     system("pause");
16     return 0;
17  }
```

```
/* prog9_4 OUTPUT---

請輸入 age[0]的值:12
請輸入 age[1]的值:54
請輸入 age[2]的值:55
age[0]=12
age[1]=54
age[2]=55
--------------------*/
```

土地vs陣列

- 土地使用：向地政事務所登記持有的街道地址作為特定用途

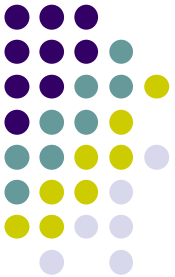
  餐廳 22巷1-2號; //登記
  22巷1號 = 庫肯花園;
  22巷2號 = Is義式餐廳;

  - 以門牌區分街道的空間
  - 地址不重複、土地位置、空間現狀 與單址相同

- 陣列宣告：由作業系統分配閒置且連續的記憶體儲存資料

  int num[2]; //宣告
  num[0] = 12400;
  num[1] = 6600;

  - 以索引區分陣列的空間
  - 名稱不重複、陣列位置、殘留值 與變數相同



陣列初值的設定 (1/2)

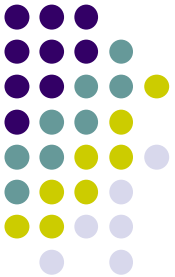
- 在宣告時就給予陣列初值，可用下面的語法

```
資料型態 陣列名稱[n]={初值 0,初值 1,...,初值 n-1};
```

- C陣列宣告及初值的設定範例

```
int day[12]={31,28,31,30,31,30,31,31,30,31,30,31};
int day[]={31,28,31,30,31,30,31,31,30,31,30,31};
int data[5]={100}; // 將陣列 data 內的所有元素值都設為 100
```

任意長度



陣列初值的設定 (2/2)

- **prog8_2是一維陣列設定初值的範例**

```
01  // prog8_2, 一維陣列的設值
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int i,a[]={15,6,8};
08     int length=sizeof(a)/sizeof(int);          // 計算陣列元素個數
09     for(i=0;i<length;i++)                      // 印出陣列的內容
10        cout << "a[" << i << "]=" << a[i] << ", ";
11     cout << endl << "array a has " << length << " elements";  // 印出 length
12     system("pause");
13     return 0;
14  }
```

int 佔幾個 byte

```
/* prog8_2 OUTPUT------

a[0]=15, a[1]=6, a[2]=8,
array a has 3 elements

----------------------*/
```

極值搜尋

下面的例子說明如何將陣列裡的最大及最小值列出

```
01  // prog8_3, 比較陣列元素值的大小
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int A[]={48,75,30,17,62};
08     int i,min=A[0],max=A[0];
09     int length=sizeof(A)/sizeof(int);    // 計算陣列元素個數
10     cout << "elements in array A are ";
11     for(i=0;i<length;i++)                // 印出陣列的內容
12     {
13        cout << 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     cout << endl << "Maximum is " << max;            // 印出最大值
20     cout << endl << "Minimum is " << min << endl;    // 印出最小值
21     system("pause");
22     return 0;
23  }
```

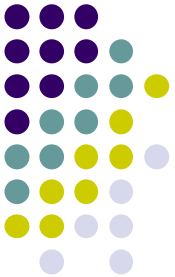
```
/* prog8_3 OUTPUT-----------------------

elements in array A are 48  75  30  17  62
Maximum is 75
Minimum is 17

-----------------------------------------*/
```

練習

- 下列陣列初始值設定方式，哪些是正確的

1. ✗
```
int arr = {1, 2, 3, 4, 5, 6};
```
2. ✗
```
int arr[5] = {1, 2, 3, 4, 5, 6};
```
3. ✗ ✓
```
int arr[10] = {1, 2, 3, 4, 5, 6};
```
4. ✗ ✓
```
int arr[10] = {1, 2.9, 3, 6.4,
5.5};
```
非整數轉整數=[1,2,3,6,5]

5. ✓
```
int arr[] = {1, 2, 3, 6};
```



陣列界限的檢查

- C++並不會檢查註標值的大小
- 當註標值超過陣列的長度時，可能造成不可預期的錯誤
- 這種錯誤是在執行時才發生的（run-time error），而不是在編譯時期發生的錯誤（compile-time error），編譯程式無法提出任何的警告訊息

土地vs陣列

- 土地使用：向地政事務所登記持有的街道地址作為特定用途

  餐廳 22巷1-2號; //登記

  22巷2號 = Is義式餐廳;

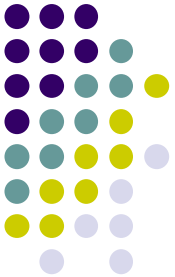
  - 22巷1號?
  - 22巷3號? 不確定是否存在

- 陣列宣告：由作業系統分配閒置且連續的記憶體儲存資料

  int num[2]; //宣告

  num[1] = 6600;

  - num[0]?
  - num[2]? 非法



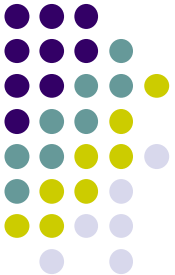
陣列界限的錯誤範例

- 一維陣列錯誤的範例

```
01  /* prog9_2, 一維陣列的基本操作(錯誤的示範) */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int i,score[4];
07
08     score[0]=78;
09     score[1]=55;
10     /* score[2]=92;  此行刻意不將 score[2] 設值 */
11     score[3]=80;
12
13     for(i=0;i<=4;i++)  /* 此行刻意將索引值超出陣列 score 的可容許範圍 */
14        printf("score[%d]=%d\n",i,score[i]);
15     system("pause");
16     return 0;
17  }
```

```
/* prog9_2 OUTPUT---

score[0]=78
score[1]=55
score[2]=51
score[3]=80
score[4]=2293600
--------------------*/
```

這兩個值都是原先留於記憶體內的殘值

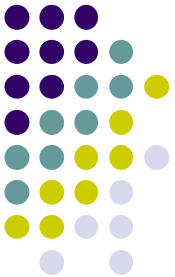


陣列界限的檢查 (1/2)

- 下面的程式裡，將陣列界限的檢查範圍加入

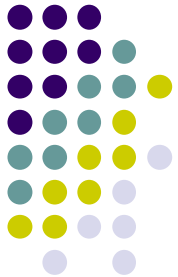
```
01  // prog8_4, 陣列的界限檢查
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  #define MAX 5          #define 識別名稱 代換標記
06  int main(void)
07  {
08     int score[MAX];
09     int i=0,num;
10     float sum=0.0f;
11     cout << "Enter 0 stopping input!!" << endl;
```

```
/* prog8_4 OUTPUT--------
Enter 0 stopping input!!
Input score:68
Input score:93
Input score:84
Input score:71
Input score:63
No more space!!
Average of all is 75.8
------------------------*/
```



陣列界限的檢查 (2/2)

```
12      do
13      {
14         if(i==MAX)              // 當 i 的值為 MAX，表示陣列已滿，即停止輸入
15         {
16            cout << "No more space!!" << endl;
17            i++;
18            break;
19         }
20         cout << "Input score:";
21         cin >> score[i];
22      }while(score[i++]>0);       // 輸入成績，輸入 0 或負數時結束
23      num=i-1;
24      for(i=0;i<num;i++)
25         sum+=score[i];           // 計算平均成績
26      cout << "Average of all is " << sum/num << endl;
27      system("pause");
28      return 0;
29   }
```



陣列資料的搜尋 (1/2)

- 在陣列中搜尋想要的資料：

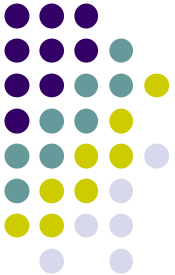
```
/* prog9_8 OUTPUT-----------
陣列 A 元素的值為:33 75 69 41 33 19
請輸入欲搜尋的整數:33
找到了! A[0]=33
找到了! A[4]=33
---------------------------*/
```

```
01  /* prog9_8, 陣列的搜尋 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define SIZE 6   /* 定義 SIZE 為 6 */
05  int main(void)
06  {
07     int i,num,flag=0;
08     int A[SIZE]={33,75,69,41,33,19};
09
10     printf("陣列 A 元素的值為:");
11     for(i=0;i<SIZE;i++)
12        printf("%d ",A[i]);              /* 印出陣列的內容 */
13
14     printf("\n 請輸入欲搜尋的整數:");
15     scanf("%d",&num);                  /* 輸入欲搜尋的整數 */
16
```

陣列資料的搜尋 (2/2)

```
17      for(i=0;i<SIZE;i++)
18         if(A[i]==num)    /* 判斷陣列元素是否與輸入值相同 */
19         {
20            printf("找到了! A[%d]=%d\n",i,A[i]);
21            flag=1;       /* 設 flag 為 1，代表有找到相同的數值 */
22         }
23      if(flag==0)
24         printf("沒有找到相同值!!\n");
25
26      system("pause");
27      return 0;
28   }
```

```
/* prog9_8 OUTPUT------------

陣列 A 元素的值為:33 75 69 41 33 19
請輸入欲搜尋的整數:33
找到了! A[0]=33
找到了! A[4]=33
----------------------------*/
```



二維陣列的宣告 (1/4)

- 二維陣列（2-dimensional array） 宣告格式

資料型態 陣列名稱 [列的個數] [行的個數];

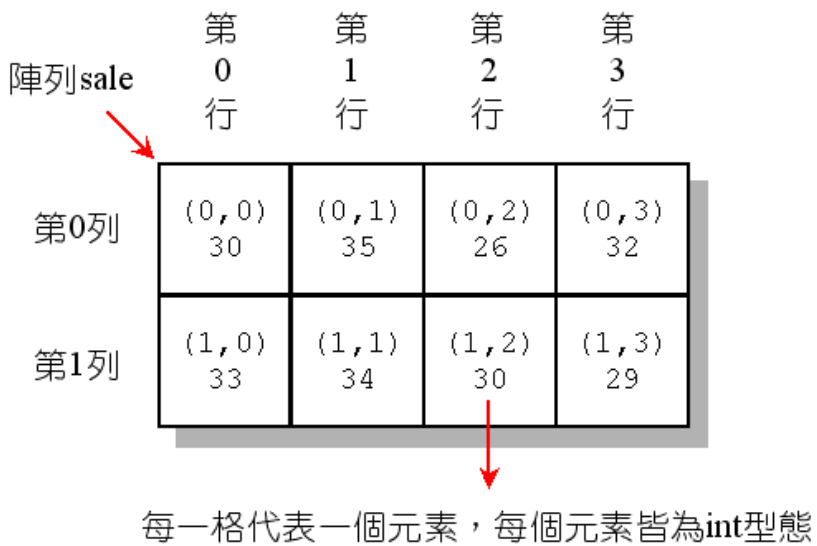
- 下面的範例都是合法的陣列宣告

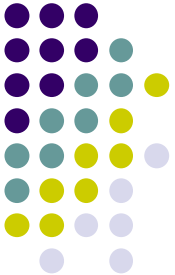
```
int data[6][5];      // 宣告整數陣列 data，元素個數為 6*5=30
float score[3][7];   // 宣告浮點數陣列 score，元素個數為 3*7=21
```

二維陣列的宣告 (2/4)

- 下表是年度銷售量，可利用二維陣列將資料儲存起來

業務員	本年度銷售量			
	第一季	第二季	第三季	第四季
1	30	35	26	32
2	33	34	30	29





二維陣列的宣告 (3/4)

- 想直接在宣告時就給予陣列初值，如下面的格式

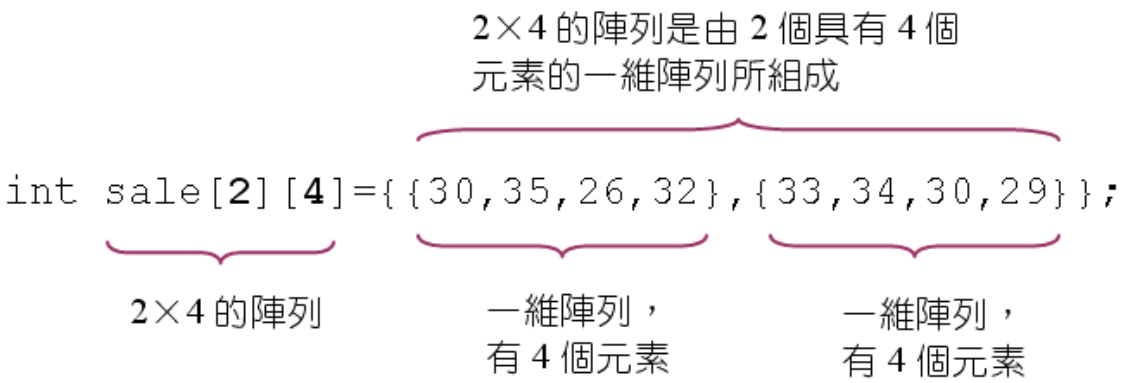
```
資料型態 陣列名稱[列的個數][行的個數]={{ 第 0 列初值 },
                                        { 第 1 列初值 },
                                        {    ...     },
                                        { 第 n 列初值 }};
```

- 下面的陣列sale宣告及初值的設定範例

```
int sale[2][4]={{30,35,26,32},     // 二維陣列的初值設定
                {33,34,30,29}};
```

二維陣列的宣告 (4/4)

- 二維陣列的初值設定可依下面的說明來設定



- 宣告二維陣列，並設定初值時，列的個數可以省略

```
int temp[][4]={{30,35,26,32},
               {33,34,30,29},
               {25,33,29,25}};
```

二維陣列元素的引用及存取

- 以二維陣列sale為例，介紹如何讀取二維陣列

```
01  // prog8_5, 二維陣列的使用
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int i,j,sum=0;
08     int sale[2][4]={{30,35,26,32},{33,34,30,29}};// 宣告陣列並設定初值
09     for(i=0;i<2;i++)                  // 輸出銷售量並計算總銷售量
10     {
11        cout << "業務員" << (i+1) << "的業績分別為 ";
12        for(j=0;j<4;j++)
13        {
14           cout << sale[i][j] << " ";
15           sum+=sale[i][j];
16        }
17        cout << endl;
18     }
19     cout << endl << "本年度總銷售量為" << sum << "輛車" << endl;
20     system("pause");
21     return 0;
22  }
```

```
/* prog8_5 OUTPUT-----------

業務員 1 的業績分別為 30 35 26 32
業務員 2 的業績分別為 33 34 30 29

本年度總銷售量為 249 輛車

----------------------------*/
```

矩陣的加法運算

$$A=\begin{bmatrix}1 & 2 & 3\\ 5 & 6 & 8\end{bmatrix};\quad B=\begin{bmatrix}3 & 0 & 2\\ 3 & 5 & 7\end{bmatrix}$$

```
01  /* prog9_10, 矩陣的相加 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define ROW 2     /* 定義 ROW 為 2 */
05  #define COL 3     /* 定義 COL 為 3 */
06  int main(void)
07  {
08     int i,j;
09     int A[ROW][COL]={{1,2,3},{5,6,8}};
10     int B[ROW][COL]={{3,0,2},{3,5,7}};
11     printf("Matrix A+B=\n");
12     for(i=0;i<ROW;i++)      /* 外層迴圈 */
13     {
14        for(j=0;j<COL;j++)   /* 內層迴圈 */
15           printf("%3d",A[i][j]+B[i][j]); /* 計算二陣列相加 */
16        printf("\n");
17     }
18     system("pause");
19     return 0;
20  }
```

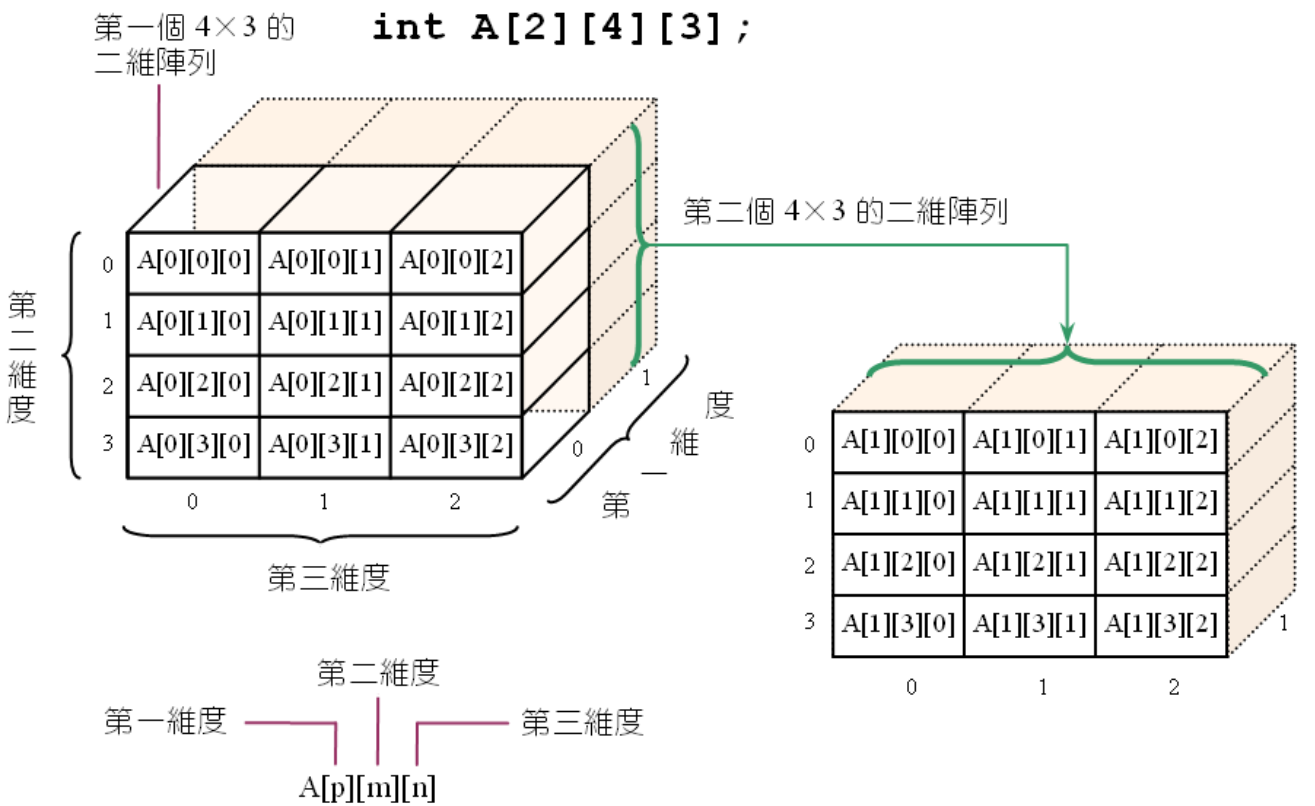
$$A+B=\begin{bmatrix}1 & 2 & 3\\ 5 & 6 & 8\end{bmatrix}+\begin{bmatrix}3 & 0 & 2\\ 3 & 5 & 7\end{bmatrix}=\begin{bmatrix}1+3 & 2+0 & 3+2\\ 5+3 & 6+5 & 8+7\end{bmatrix}=\begin{bmatrix}4 & 2 & 5\\ 8 & 11 & 15\end{bmatrix}$$

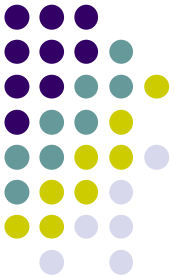
```
/* prog9_10 OUTPUT---

Matrix A+B=
  4  2  5
  8 11 15
--------------------*/
```

多維陣列

- 要宣告一個2×4×3的陣列A，可以利用下面的語法

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





三維陣列中搜尋最大值 (1/2)

- 下面的範例說明如何在三維陣列裡，找出元素的最大值

```
01  // prog8_6, 三維陣列的使用
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int A[2][4][3]={{{21,32,65},          // 宣告陣列並設定初值
08                      {78,94,76},
09                      {79,44,65},
10                      {89,54,73}},
11                     {{32,56,89},
12                      {43,23,32},
13                      {32,56,78},
14                      {94,78,45}}};
15     int i,j,k,max=A[0][0][0];             // 設定 max 為 A 陣列的第一個元素
16
```

設定 2×4×3 陣列的初值

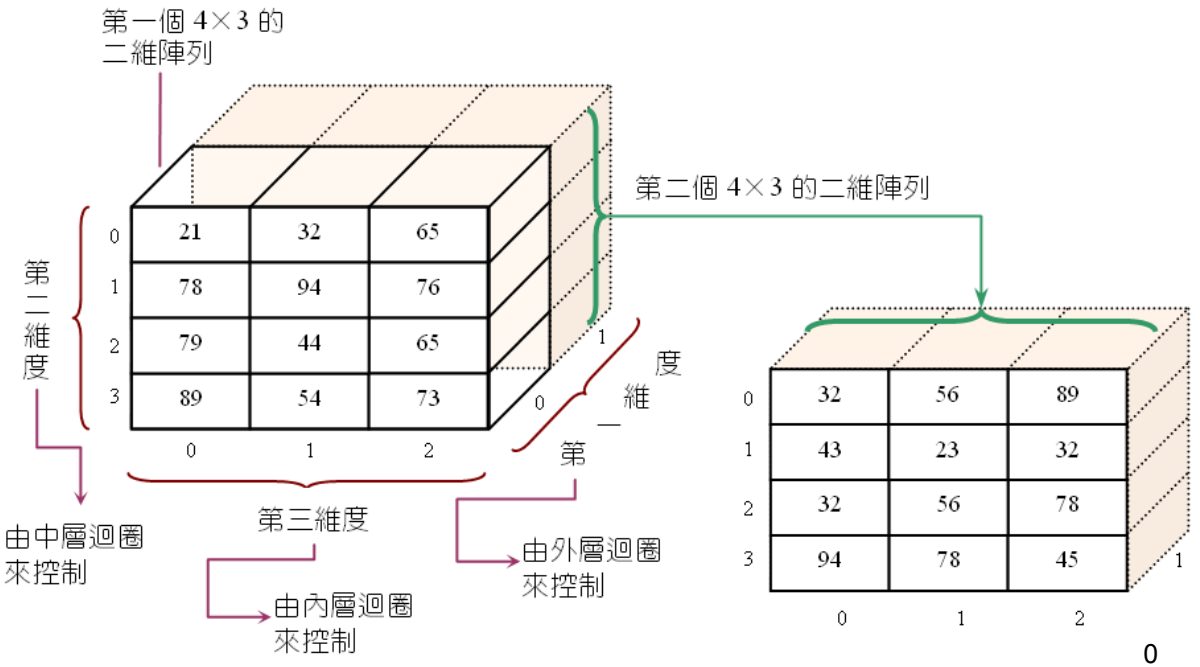
```
/* prog8_6 OUTPUT---
max=94
--------------------*/
```

三維陣列中搜尋最大值 (1/2)

```
17      for(i=0;i<2;i++)
18         for(j=0;j<4;j++)
19            for(k=0;k<3;k++)
20               if(max<A[i][j][k])
21                  max=A[i][j][k];
22      cout << "max=" << max << endl;      // 印出陣列的最大值
23
24      system("pause");
25      return 0;
26   }
```

利用三個 for 迴圈找出陣列的最大值

```
/* prog8_6 OUTPUT---
max=94
--------------------*/
```



三維陣列的初始化

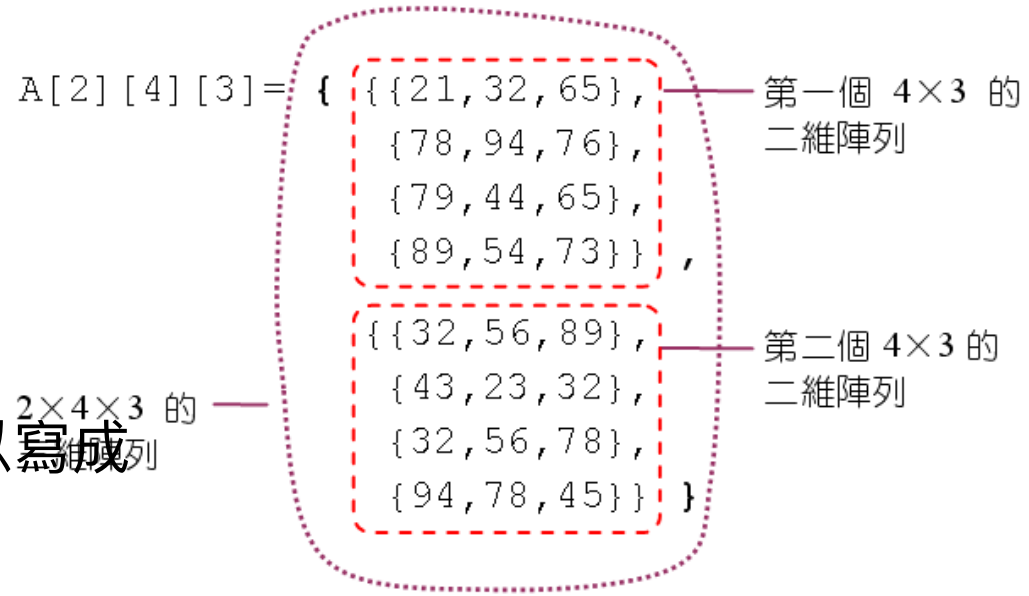
- 三維陣列A的第一個4×3的二維陣列為

```
{{21,32,65},
 {78,94,76},
 {79,44,65},
 {89,54,73}}
```

第二個4×3的二維陣列為

```
{{32,56,89},
 {43,23,32},
 {32,56,78},
 {94,78,45}}
```

因此2×4×3的三維陣列可以寫成



2×4×3 的三維陣列 ＝{ 4×3 的二維陣列，4×3 的二維陣列 }

-The End-