第九章
陣列與字串

一維、二維與多維陣列

傳遞陣列給函數

字串以及字串相關的函數

字元陣列

一維陣列

- 陣列是相同型態之元素所組成的集合
- 在 C 語言中，陣列使用前必須先宣告：

一維陣列的宣告格式

```
資料型態  陣列名稱[個數];
```

- 一維陣列宣告的範例：

```
int score[4];        /* 宣告整數陣列score，可存放4個元素 */
float temp[7];       /* 宣告浮點數陣列temp，可存放7個元素 */
char name[12];       /* 宣告字元陣列name，可存放12個元素 */
```


陣列的索引值

- 陣列中的元素是以索引值來標示存放的位置
- 陣列索引值的編號由 0 開始

```
int score[4];
```

簡單的範例 (1/2)

- 一維陣列的基本操作：

```
/* prog9_1 OUTPUT---

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

```
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  }
```

簡單的範例 (2/2)

- 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

一維陣列初值的設定

- 一維陣列初值的設定格式：

一維陣列初值設定的格式

```
資料型態  陣列名稱[個數n]={初值1,初值2,…,初值n};
```

- 初值設定的範例：
  - `int score[4]={78,55,92,80};`
  - `int score[]={60,75,48,92};   /* 省略元素的個數 */`
  - `int data[5]={0};`

sizeof 運算子

- 查詢陣列所佔的記憶空間

查詢整個陣列所佔的位元組

```
sizeof(陣列名稱)    /* 查詢陣列所佔的位元組 */
```

```
01  /* prog9_3, 查詢陣列所佔的記憶空間 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     double data[4];    /* 宣告有 4 個元素的 double 型態陣列 */
07     printf("陣列元素所佔的位元組:%d\n",sizeof(data[0]));
08     printf("整個陣列所佔的位元組:%d\n",sizeof(data));
09     printf("陣列元素的個數:%d\n",sizeof(data)/sizeof(double));
10     system("pause");
11     return 0;
12  }
```

```
/* prog9_3 OUTPUT---
陣列元素所佔的位元組:8
整個陣列所佔的位元組:32
陣列元素的個數:4
--------------------*/
```

sizeof("中文字串"); // Big5: 9, UTF-8: 13

陣列資料的拷貝

- 陣列資料的拷貝

```
01 #include <string.h> // memcpy()
02 int main(void)
03 {
04      int score1[]={78,55,92,80}, score2[4], i;
05      score2 = score1;   // compile-time error
06      for (i=0; i<sizeof(score1)/sizeof(int); i++)
07          score2[i] = score1[i];
08      memcpy(score2, score1, sizeof(score1));
09 }
```

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


陣列的應用－最大與最小值

```
01  /* prog9_5, 比較陣列元素值的大小 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int A[5]={74,48,30,17,62};
07     int i,min,max;
08     min=max=A[0];      /* 將 max 與 min 均設為陣列的第一個元素 */
09     for(i=0;i<5;i++)
10     {
11        if(A[i]>max)    /* 判斷 A[i]是否大於 max */
12           max=A[i];
13        if(A[i]<min)    /* 判斷 A[i]是否小於 min */
14           min=A[i];
15     }
16     printf("陣列裡元素的最大值為%d\n",max);
17     printf("陣列裡元素的最小值為%d\n",min);
18     system("pause");
19     return 0;
20  }
```

```
/* prog9_5 OUTPUT---

陣列裡元素的最大值為 74
陣列裡元素的最小值為 17
--------------------*/
```


輸入個數未定的資料

```
01  /* prog9_6, 輸入未定個數的資料到陣列裡 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define MAX 10
05  int main(void)
06  {
07     int score[MAX];
08     int i=0,num;
09     int sum=0;
10     printf("請輸入成績，要結束請輸入 0:\n");
11     do
12     {
13        printf("請輸入成績:");
14        scanf("%d",&score[i]);
15     }while(score[i++]>0);          /* 輸入成績，輸入 0 時結束 */
16     num=i-1;
17     for(i=0;i<num;i++)
18        sum+=score[i];              /* 計算平均成績 */
19     printf("平均成績為 %.2f\n",(float)sum/num);
20     system("pause");
21     return 0;
22  }
```

```
/* prog9_6 OUTPUT---

請輸入成績，要結束請輸入 0:
請輸入成績:70
請輸入成績:80
請輸入成績:60
請輸入成績:90
請輸入成績:0
平均成績為 75.00
--------------------*/
```


陣列界限的檢查 (1/2)

- C語言不會自動檢查陣列界限（可增快執行速度）
- 下面的範例加入了陣列界限的檢查：

```
01  /* prog9_7, 陣列的界限檢查 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define MAX 5
05  int main(void)
06  {
07     int score[MAX];
08     int i=0,num;
09     float sum=0.0f;
10     printf("請輸入成績，要結束請輸入 0:\n");
```

```
/* prog9_7 OUTPUT---

請輸入成績，要結束請輸入 0:
請輸入成績:60
請輸入成績:50
請輸入成績:70
請輸入成績:80
請輸入成績:90
陣列空間已使用完畢!!
平均成績為 70.00
--------------------*/
```

陣列界限的檢查 (2/2)

```
11      do
12      {
13         if(i==MAX)  /* 當 i 的值為 MAX 時，表示陣列已滿，即停止輸入 */
14         {
15            printf("陣列空間已使用完畢!!\n");
16            i++;      /* 此行先將 i 值加 1，因為 23 行會把 i 的值減 1 掉 */
17            break;
18         }
19         printf("請輸入成績:");
20         scanf("%d",&score[i]);
21      }while(score[i++]>0); /* 輸入 0 時結束 */
22      num=i-1;
23      for(i=0;i<num;i++)
24         sum+=score[i];       /* 計算平均成績 */
25      printf("平均成績為 %.2f\n",sum/num);
26
27      system("pause");
28      return 0;
29   }
```

```
/* prog9_7 OUTPUT---
請輸入成績，要結束請輸入 0:
請輸入成績:60
請輸入成績:50
請輸入成績:70
請輸入成績:80
請輸入成績:90
陣列空間已使用完畢!!
平均成績為 70.00
--------------------*/
```


陣列資料的搜尋 (1/2)

- 在陣列中搜尋想要的資料：

```
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
```

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


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

- 二維陣列的宣告

二維陣列的宣告格式

資料型態 陣列名稱[列的個數][行的個數];

- 二維陣列宣告的範例：

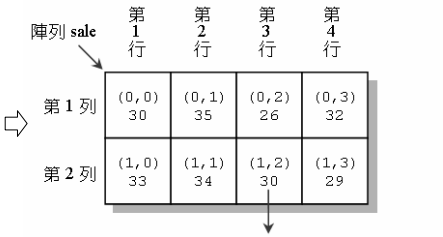
```
int data[10][5];     /* 可存放10列5行個整數 */
float score[4][3];   /* 可存放4列3行個浮點數 */
```

表格與二維陣列

- 二維的表格很適合用陣列來儲存：　業績以二維陣列表示

表 9.2.1 業務員於 2004 年每季的銷售業績

業務員	2004 年銷售量			
	第一季	第二季	第三季	第四季
1	30	35	26	32
2	33	34	30	29



每一格代表一個元素，每個元素皆為 int 型態

可以不填

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]={{30,35,26,32},{33,34,30,29}};
```

2×4 的陣列　一維陣列，有 4 個元素　一維陣列，有 4 個元素

```
int sale[2][4]={{30,35,26,32},
                {33,34,30,29}};
```

寫成這樣較易理解

二維陣列元素的存取 (1/2)

- 利用巢狀迴圈依序輸入二維陣列的元素：

```
01  /* prog9_9, 二維陣列的輸入輸出 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int i,j,sale[2][4],sum=0;
07
08     for(i=0;i<2;i++)
09        for(j=0;j<4;j++)
10        {
11           printf("業務員%d 的第%d 季業績:",i+1,j+1);
12           scanf("%d",&sale[i][j]);   /* 輸入銷售量 */
13        }
14
```

```
/* prog9_9 OUTPUT----------
業務員 1 的第 1 季業績:30
業務員 1 的第 2 季業績:35
業務員 1 的第 3 季業績:26
業務員 1 的第 4 季業績:32
業務員 2 的第 1 季業績:33
業務員 2 的第 2 季業績:34
業務員 2 的第 3 季業績:30
業務員 2 的第 4 季業績:29
***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
2004 年總銷售量為 249 部車
----------------------------*/
```


二維陣列元素的存取 (2/2)

```
15     printf("***Output***");
16     for(i=0;i<2;i++)                  /* 輸出銷售量並計算總銷售量 */
17     {
18        printf("\n業務員%d 的業績分別為",i+1);
19        for(j=0;j<4;j++)
20        {
21           printf("%d  ",sale[i][j]);
22           sum+=sale[i][j];
23        }
24     }
25     printf("\n2004 年總銷售量為%d 部車\n",sum);
26
27     system("pause");
28     return 0;
29  }
```

```
/* prog9_9 OUTPUT----------
業務員 1 的第 1 季業績:30
業務員 1 的第 2 季業績:35
業務員 1 的第 3 季業績:26
業務員 1 的第 4 季業績:32
業務員 2 的第 1 季業績:33
業務員 2 的第 2 季業績:34
業務員 2 的第 3 季業績:30
業務員 2 的第 4 季業績:29
***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
2004 年總銷售量為 249 部車
----------------------------*/
```


矩陣的加法運算

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

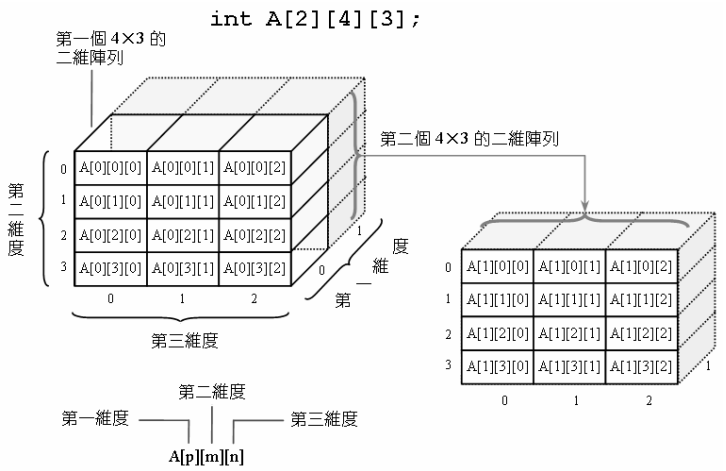
$$A+B=\begin{bmatrix}1 & 2 & 3\\ 5 & 6 & 8\end{bmatrix}+\begin{bmatrix}3 & 0 & 2\\ 3 & 5 & 7\end{bmatrix}=\begin{bmatrix}1+3 & 2+0 & 3+2\\ 5+3 & 6+5 & 8+7\end{bmatrix}=\begin{bmatrix}4 & 2 & 5\\ 8 & 11 & 15\end{bmatrix}$$

```
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]); /*
16        printf("\n");
17     }
18     system("pause");
19     return 0;
20  }
```

```
/* prog9_10 OUTPUT---
Matrix A+B=
  4  2  5
  8 11 15
----------------------*/
```

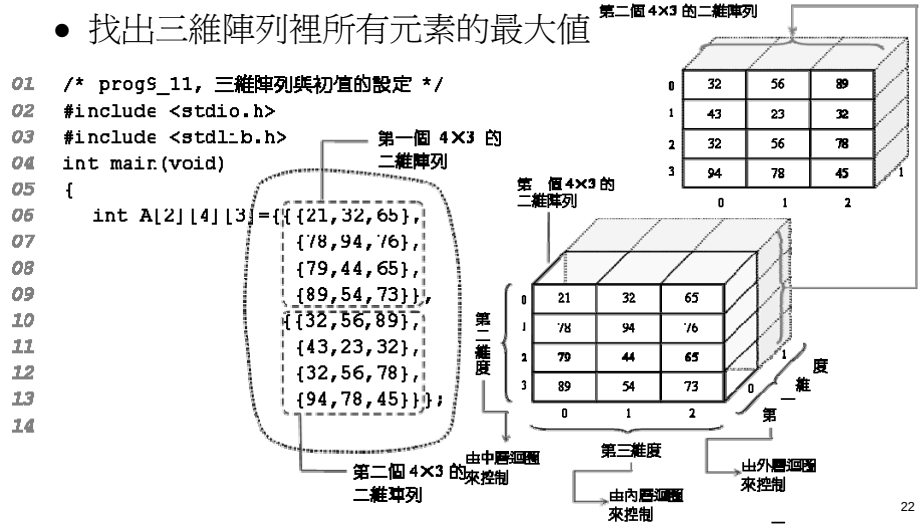
多維陣列

- 三維陣列的結構（以2x4x3的陣列爲例）：



三維陣列的使用 (1/2)

- 找出三維陣列裡所有元素的最大值




```
01  /* prog9_11, 三維陣列與初值的設定 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int A[2][4][3]={{{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
```


三維陣列的使用 (2/2)

```
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<2;i++)            /* 外層迴圈 */
18        for(j=0;j<4;j++)         /* 中層迴圈 */
19           for(k=0;k<3;k++)      /* 內層迴圈 */
20              if(max<A[i][j][k])
21                 max=A[i][j][k];
22
23     printf("max=%d\n",max);     /* 印出陣列的最大值 */
24     system("pause");
25     return 0;
26  }
```

利用三個 for 迴圈找出陣列的最大值

```
/* prog9_11 OUTPUT---

max=94
----------------------*/
```


傳遞陣列到函數

函數常常代表一個動作,函數的參數就是這個動作的標的物

- 以一維陣列爲引數來傳遞陣列的格式：

傳遞一維陣列到函數

```
傳回值型態 函數名稱(資料型態 陣列名稱[]);  /* 原型 */
int main(void)
{
    資料型態 陣列名稱[個數];
        ...
    函數名稱(陣列名稱);
       ...
}
傳回值型態 函數名稱(資料型態 陣列名稱[])
{
    ...
}
```

陣列大小可以省略

傳遞一維陣列的範例

```
01  /* prog9_12, 傳遞一維陣列到函數裡 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define SIZE 4
05  void show(int arr[]);          /* 宣告函數 show()的原型 */
06  int main(void)
07  {
08     int A[SIZE]={5,3,6,1};      /* 設定陣列 A 的初值 */
09     printf("陣列的內容為: ");
10     show(A);                    /* 呼叫函數 show()  */
11     system("pause");
12     return 0;
13  }
14  void show(int arr[])           /* 函數 show()的定義 */
15  {
16     int i;
17     for(i=0;i<SIZE;i++)
18        printf("%d ",arr[i]);    /* 印出陣列內容 */
19     printf("\n");
20  }
```

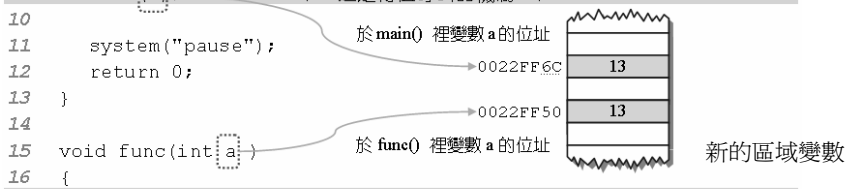
```
/* prog9_12 OUTPUT---
陣列的內容為: 5 3 6 1
--------------------*/
```


傳遞數值到函數 (傳值呼叫)

- 傳遞整數到函數 func() 裡的範例 （傳值）：

```
01  /* prog9_13, 印出變數的位址 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  void func(int);
05  int main(void)
06  {
07     int a=13;
08     printf("於 main()裡,a=%d,a 的位址=%p\n",a,&a);
09     func(a);              /* 這是傳值呼叫的機制 */
10
11     system("pause");
12     return 0;
13  }
14
15  void func(int a)
16  {
17     printf("於 func()裡,a=%d,a 的位址為=%p\n",a,&a);
18  }
```

```
/* prog9_13 OUTPUT-------------
於 main()裡,a=13,a 的位址=0022FF6C
於 func()裡,a=13,a 的位址=0022FF50
--------------------------------*/
```

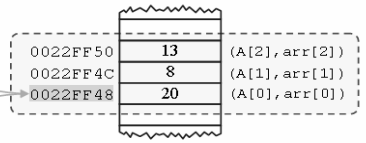


傳遞位址到函數 (傳址呼叫)

```
01  /* prog9_14, 印出陣列的位址 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define SIZE 3
05  void func(int arr[]);
06  int main(void)
07  {
08     int i,A[SIZE]={20,8,13};
09     printf("在 main()裡，陣列 A 元素的位址為\n");
10     for(i=0;i<SIZE;i++)
11        printf("A[%d]=%2d,位址為%p\n",i,A[i],&A[i]);
12     func(A);                    /* 這是傳址呼叫的機制 */
13     system("pause");
14     return 0;
15  }
16  void func(int arr[])
17  {
18     int i;
19     printf("\n 在 func()裡，陣列 arr 元素的位址為\n");
20     for(i=0;i<SIZE;i++)
21        printf("arr[%d]=%2d,位址為%p\n",i,arr[i],&arr[i]);
22  }
```

arr[i] 和 main 裡面的變數 A[i] 是相同的

```
/* prog9_14 OUTPUT--------
在 main()裡，陣列 A 元素的位址為
A[0]=20,位址為 0022FF48
A[1]= 8,位址為 0022FF4C
A[2]=13,位址為 0022FF50

在 func()裡，陣列 arr 元素的位址為
arr[0]=20,位址為 0022FF48
arr[1]= 8,位址為 0022FF4C
arr[2]=13,位址為 0022FF50
---------------------------*/
```



陣列的位址

- C語言是以陣列第一個元素的位址當成是陣列的位址
- 陣列名稱本身就是存放陣列位址的const變數

```
01  /* prog9_15, 印出陣列的位址 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define SIZE 3
05  int main(void)
06  {
07     int i,A[SIZE]={20,8,13};
08     for(i=0;i<SIZE;i++)
09        printf("A[%d]=%2d,位址為%p\n",i,A[i],&A[i]);
10     printf("陣列 A 的位址=%p\n",A);
11     system("pause");
12     return 0;
13  }
```

```
/* prog9_15 OUTPUT----
A[0]=20,位址=0022FF48
A[1]= 8,位址=0022FF4C
A[2]=13,位址=0022FF50
陣列 A 的位址=0022FF48
-------------------------*/
```

傳遞陣列到函數的應用 (1/2)

- 於函數裡變更陣列元素的值：

```
01  /* prog9_16, 於函數內更改陣列元素的值 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define SIZE 4
05  void show(int arr[]);
06  void add2(int arr[]);
07
08  int main(void)
09  {
10     int A[SIZE]={5,3,6,1};
11     printf("呼叫 add2()前,陣列的內容為: ");
12     show(A);            /* 呼叫函數 show() */
13     add2(A);            /* 呼叫函數 add2() */
14     printf("呼叫 add2()後,陣列的內容為: ");
15     show(A);            /* 呼叫函數 show() */
16     system("pause");
17     return 0;
18  }
```

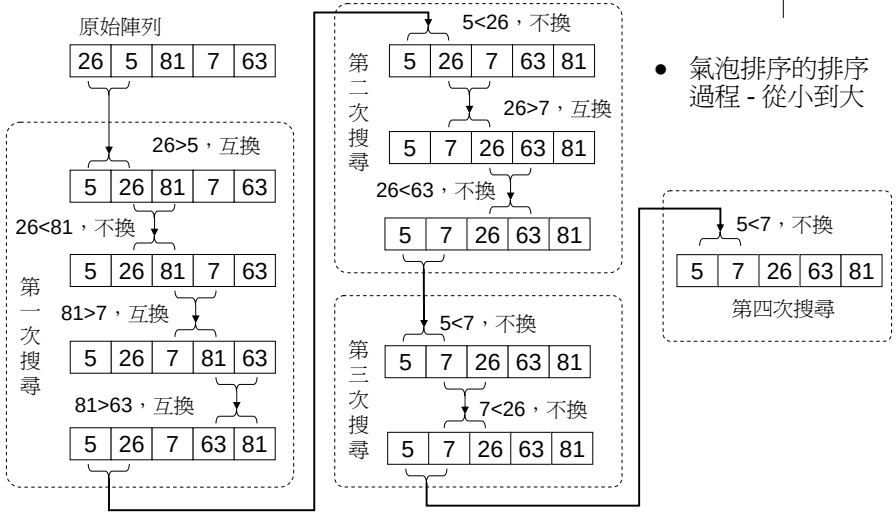
```
/* prog9_16 OUTPUT------------

呼叫 add()前,陣列的內容為: 5 3 6 1
呼叫 add()後,陣列的內容為: 7 5 8 3
------------------------------*/
```


傳遞陣列到函數的應用 (2/2)

```
19  void show(int arr[])
20  {
21     int i;
22     for(i=0;i<SIZE;i++)    /* 印出陣列內容 */
23        printf("%d ",arr[i]);
24     printf("\n");
25  }
26  void add2(int arr[])
27  {
28     int i;
29     for(i=0;i<SIZE;i++)
30        arr[i]+=2;
31  }
```

```
/* prog9_16 OUTPUT------------

呼叫 add()前,陣列的內容為: 5 3 6 1
呼叫 add()後,陣列的內容為: 7 5 8 3
------------------------------*/
```


一維陣列的應用 - 氣泡排序法



- 氣泡排序的排序過程 - 從小到大

氣泡排序法的程式碼 (1/2)

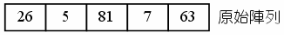
```
01  /* prog9_17, 氣泡排序法 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define SIZE 5
05  void show(int a[]), bubble(int a[]);
06  int main(void)
07  {
08     int data[SIZE]={26,5,81,7,63};
09
10     printf("排序前...\n");
11     show(data);                /* 印出陣列內容 */
12     bubble(data);              /* 呼叫 bubble()函數 */
13     printf("排序後...\n");
14     show(data);                /* 印出陣列內容 */
15     system("pause");
16     return 0;
17  }
18  void show(int a[])
19  {
20     int i;
21     for(i=0;i<SIZE;i++)
22        printf("%d ",a[i]);
23     printf("\n");
24  }
```

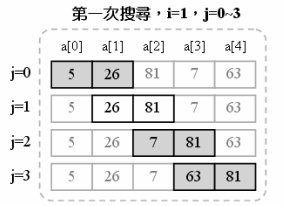
```
/* prog9_17 OUTPUT---

排序前...
26 5 81 7 63
排序後...
5 7 26 63 81
----------------------*/
```

氣泡排序法的程式碼 (2/2)

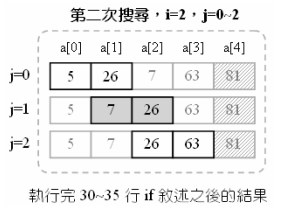
```
25  void bubble(int a[])
26  {
27     int i,j,temp;
28     for(i=1;i<SIZE;i++)
29        for(j=0;j<(SIZE-i);j++)
30           if(a[j]>a[j+1])
31           {
32              temp=a[j];
33              a[j]=a[j+1];
34              a[j+1]=temp;
35           }
36  }
```

如果 a[j]>a[j+1]，則元素的值互換



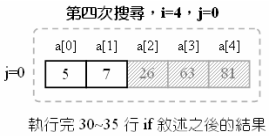


執行完 30~35 行 if 敘述之後的結果



執行完 30~35 行 if 敘述之後的結果

執行完 30~35 行 if 敘述之後的結果



執行完 30~35 行 if 敘述之後的結果

氣泡排序法的改良版

- 利用 flag 控制外層迴圈搜尋的次數：

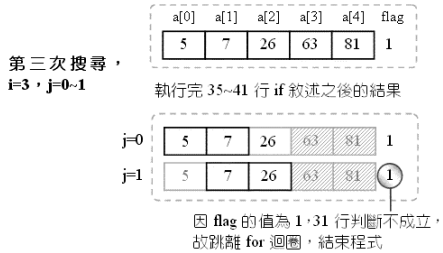
```
26  void bubble2(int a[])
27  {
28     int i,j,temp;
29
30     int flag=0;
31     for(i=1;(i<SIZE)&&(!flag);i++)
32     {
33        flag=1;
34        for(j=0;j<(SIZE-i);j++)
35           if(a[j]>a[j+1])
36           {
37              temp=a[j];
38              a[j]=a[j+1];
39              a[j+1]=temp;
40              flag=0;
41           }
42     }
43  }
```

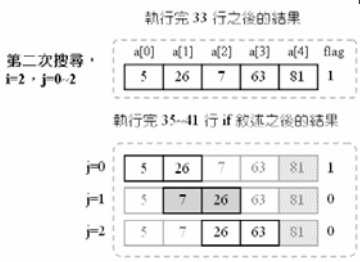
如果在某一個外層迴圈 i 中內層迴圈完全沒有交換的動作就表示已經排好了，剩下的 i+1 到 SIZE-1 次迴圈可以省略

本範例爲prog9_18, 其中 main() 函數與show() 函數已省略

```
/* prog9_18 OUTPUT---

Before process...
26 5 81 7 63
After process...
5 7 26 63 81
---------------------*/
```


改良版的圖說






```
30     int flag=0;
31     for(i=1;(i<SIZE)&&(!flag);i++)
32     {
33        flag=1;
34        for(j=0;j<(SIZE-i);j++)
35           if(a[j]>a[j+1])
36           {
37              temp=a[j];
38              a[j]=a[j+1];
39              a[j+1]=temp;
40              flag=0;
41           }
42     }
```


傳遞二維與多維陣列

- 傳遞二維陣列的格式：

二維陣列的宣告格式

```
傳回值型態 函數名稱(資料型態 陣列名稱[][行的個數]);    /* 原型 */
int main(void)
{
   資料型態 陣列名稱[列的個數][行的個數];
    ...
   函數名稱(陣列名稱);
    ...
}
傳回值型態 函數名稱(資料型態 陣列名稱[] [行的個數] )
{
    ...
}
```

必須填入行的個數

必須填入行的個數

傳遞二維陣列的範例 (1/2)

- 尋找二維陣列的最大值與最小值 ：

```
01  /* prog9_19, 尋找二維陣列的最大值與最小值 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define ROW 4
05  #define COL 3
06  void search(int a[][COL],int b[]);     /* search() 函數的原型 */
07  int main(void)
08  {
09     int a[ROW][COL]= {{26, 5, 7},
10                       {10, 3,47},
11                       { 6,76, 8},
12                       {40, 4,32}};
13     int i,j,b[2];
14     printf("二維陣列內的元素:\n");
15     for(i=0;i<ROW;i++)
16     {
17        for(j=0;j<COL;j++)
18           printf("%02d ",a[i][j]);
19        printf("\n");
20     }
```

```
/* prog9_19 OUTPUT---

二維陣列內的元素:
26 05 07
10 03 47
06 76 08
40 04 32
陣列的最大值=76
陣列的最小值=03
----------------------*/
```


傳遞二維陣列的範例 (2/2)

```
21     search(a,b);                          /* 呼叫 search()函數 */
22     printf("陣列的最大值=%02d\n",b[0]);    /* 印出陣列的最大值 */
23     printf("陣列的最小值=%02d\n",b[1]);    /* 印出陣列的最小值 */
24     system("pause");
25     return 0;
26  }
27  void search(int arr[][COL],int p[])  /* 自訂函數 search() */
28  {
29     int i,j;
30     p[0]=p[1]=arr[0][0];      /* 將 p[0]與 p[1]均設為 arr[0][0] */
31     for(i=0;i<ROW;i++)
32        for(j=0;j<COL;j++)
33        {
34           if(p[0]<arr[i][j])   /* 尋找最大值 */
35              p[0]=arr[i][j];
36           if(p[1]>arr[i][j])   /* 尋找最小值 */
37              p[1]=arr[i][j];
38        }
39  }
```

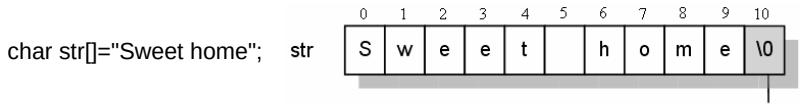
```
/* prog9_19 OUTPUT---

二維陣列內的元素:
26 05 07
10 03 47
06 76 08
40 04 32
陣列的最大值=76
陣列的最小值=03
----------------------*/
```


字串的宣告與初值的設定

- 字元以單引號包圍，而字串則是以雙引號包圍：
  - 'a' /* 這是字元常數 a */
  - "a" /* 這是字串常數 a */
  - "Sweet home" /* 這是字串常數 Sweet home */
- 下面是字串宣告的語法：

字串宣告的語法

```
char 字元陣列名稱[陣列大小] = 字串常數;
```



字元與字串之比較

- 字元與字串之比較的範例：

```
01  /* prog9_20, 印出字元及字串的長度 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     char ch='a';                /* 宣告字元變數 ch */
07     char str1[]="a";            /* 宣告字串變數 str1 */
08     char str2[]="Sweet home";   /* 宣告字串變數 str2 */
09
10     printf("字元 ch 佔了%d 個位元組\n",sizeof(ch));
11     printf("字串 str1 佔了%d 個位元組\n",sizeof(str1));
12     printf("字串 str2 佔了%d 個位元組\n",sizeof(str2));
13
14     system("pause");
15     return 0;
16  }
```

```
/* prog9_20 OUTPUT---

字元 ch 佔了 1 個位元組
字串 str1 佔了 2 個位元組
字串 str2 佔了 11 個位元組
----------------------*/
```

字串的輸入與輸出函數

- gets() 與 puts() 的格式：

gets() 的格式

```
char *gets(字元陣列名稱);
```

讀入一整列, 換列字元 '\n' 會換成 '\0'

puts() 的格式

```
puts(字元陣列名稱);
```

或者是

```
puts(字串常數);
```

字串常數是一個陣列

gets() 與 puts() 函數

- gets() 與 puts() 的使用範例：

```
/* prog9_21 OUTPUT---
What's your name?
David Young
Hi!
David Young
How are you?
------------------------*/
```

```
01  /* prog9_21, 輸入及印出字串 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     char name[15];     /* 宣告字元陣列 name */
07
08     puts("What's your name?");
09     gets(name);        /* 利用 gets()讀入字串，並寫入字元陣列 name 裡 */
10     puts("Hi!");
11     puts(name);        /* 印出字元陣列 name 的內容 */
12     puts("How are you?");
13     system("pause");
14     return 0;
15  }
```

"Hi !"[3]

大小寫的轉換 (1/2)

- 大小寫的轉換範例：

```
01  /* prog9_22, 將字串裡小寫字母轉換成大寫 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  void toUpper(char s[]);   /* 宣告函數 toUpper()的原型 */
05  int main(void)
06  {
07     char str[15];          /* 宣告可容納 15 個字元的陣列 str */
08
09     printf("請輸入一個字串: ");
10     gets(str);             /* 輸入字串 */
11     toUpper(str);          /* 呼叫 toUpper() 函數 */
12     printf("轉換成大寫後: %s\n",str);      /* 印出 str 字串的內容 */
13
14     system("pause");
15     return 0;
16  }
17
```

```
/* prog9_22 OUTPUT----------
請輸入一個字串: Happy Birthday
轉換成大寫後: HAPPY BIRTHDAY
---------------------------*/
```


大小寫的轉換 (2/2)

```
18  void toUpper(char s[])
19  {
20     int i=0;
21     while(s[i]!='\0')       /* 如果 s[i] 不等於\0，則執行下面的敘述 */
22     {
23        if(s[i]>=97 && s[i]<=122) /* 如果是小寫字母 */
24           s[i]=s[i]-32;     /* 把小寫字母的 ASCII 碼減 32，變成大寫 */
25        i++;
26     }
27  }
```

```
if (s[i]>='a' && s[i]<='z')
   s[i] = s[i] – 'a' + 'A';
```

字串陣列

- 字串陣列的宣告與初值設定

字串陣列的宣告

```
char 字元陣列名稱[字串的個數][字串長度];
```

宣告字串陣列，並設定初值

```
char 字元陣列名稱[字串的個數][字串長度]=
        {"字串常數1", "字串常數2",..., "字串常數n"};
```

```
e.g.
  char customer[6][15];
  char S[3][10]={"Tom","Lily","James Lee"};
```


字串陣列元素的存取

- 字串變數於記憶體內的儲存方式：

```
/* prog9_23 OUTPUT----------
S[0]=Tom
S[1]=Lily
S[2]=James Lee

S[0]=0253FDB8
address of S[0][0]=0253FDB8

S[1]=0253FDC2
address of S[1][0]=0253FDC2

S[2]=0253FDCC
address of S[2][0]=0253FDCC
--------------------------*/
```

```
01  /* prog9_23,字串陣列 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     char S[3][10]= {"Tom","Lily","James Lee"};
07     int i;
08     for(i=0;i<3;i++)
09        printf("S[%d]=%s\n",i,S[i]);    /* 印出字串陣列內容 */
10     printf("\n");
11     for(i=0;i<3;i++)   /* 印出字串陣列元素的位址 */
12     {
13        printf("S[%d]=%p\n",i,S[i]);
14        printf("address of S[%d][0]=%p\n\n",i,&S[i][0]);
15     }
16     system("pause");
17     return 0;
18  }
```

S[i]

複製字串陣列

```
#include <string.h>
…
for (i=0; i<MAX; i++)
    strncpy(arr2[i], arr1[i], sizeof(arr2[i])-1);

strlen(), strncat(), strncmp(), …
```

```
01  /* prog9_24, 字串陣列的複
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define MAX 3
05  #define LENGTH 10
06  int main(void)                    ----------*/
07  {
08     char arr1[MAX][LENGTH]={"Tom","Lily","James Lee"};
09     char arr2[MAX][LENGTH];
10     int i,j;
11     for(i=0;i<MAX;i++)      /* 將 arr1 的內容複製到 arr2 中 */
12     {
13        for(j=0;j<LENGTH;j++)
14           if(arr1[i][j]=='\0')   /* 如果遇到「\0」,代表讀到字串結束 */
15              break;          /* 此行的 break 敘述會跳到第 19 行執行 */
16           else
17              arr2[i][j]=arr1[i][j];
18        arr2[i][j]='\0';
19     }
20     for(i=0;i<MAX;i++)
21        printf("arr2[%d]=%s\n",i,arr2[i]); /* 印出陣列 arr2 的內容 */
22     system("pause");
23     return 0;
24  }
```