

程式設計 (**Programming**)

真理大學 資訊工程系 吳汶涓老師

CH06 陣列(array)



本章綱要

6-1 簡介

6-2 陣列

6-3 定義陣列

6-4 使用陣列的例子

6-5 傳遞陣列給函式

6-6 陣列的排序

6-7 範例研究：使用陣列來計算平均數、眾數等

6-8 搜尋陣列

6-9 多維陣列

6-10 可變長度陣列

6-11 安全程式設計

6.1 簡介

■ 陣列 (array)

- 由**相同型別**的相關**資料**項所組成的資料結構
- 屬**靜態的結構**：程式執行期間大小並不會改變
- 第12章將會介紹動態的資料結構 (串列、佇列、堆疊和樹等等)

- 範例：由使用者輸入5個數字比大小
需要宣告 5 個變數

```
int a, b, c, d, e;  
scanf("%d %d %d %d %d", &a, &b, &c, &d, &e);  
int max = a;  
max = (max < b)? b: max;  
...
```

以上作法**程式碼變多、增加程式執行時間**

6.2 陣列

■ 陣列

- 一組連續的記憶體位置
- 相同名稱以及相同型別

■ 引用陣列中的某個元素，要指定

- 陣列名稱
- 位置編號

■ 格式

`arrayname[position]`

- 第一個元素在位置 0
- 具有n個元素的陣列，陣列為 `c[0]`, `c[1]`, `c[2]`, ..., `c[n-1]`

<code>c[0]</code>	-45
<code>c[1]</code>	6
<code>c[2]</code>	0
<code>c[3]</code>	72
<code>c[4]</code>	1543
<code>c[5]</code>	-89
<code>c[6]</code>	0
<code>c[7]</code>	62
<code>c[8]</code>	-3
<code>c[9]</code>	1
<code>c[10]</code>	6453
<code>c[11]</code>	78

陣列名稱 (請注意，此陣列中的所有元素都具有相同的名稱，也就是c)

■ 陣列元素如同一般的變數

- 指定值： `c[0] = 0;`
- 輸出值： `printf("%d", c[0]/2);`
- 下標(index)可執行運算 ($x = 3$)
`c[5-2] = 0;`
`c[3] = 0;`
`c[x] = 0;`

■ 注意：陣列的下標開始為 0

c[0]	-45
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78

元素在陣列c中的位置編號

圖 6.1 含有 12 個元素的陣列

6.3 定義陣列

■ 當定義陣列時，需指定：

- 名稱
- 陣列型別
- 元素個數
- 範例:

`int c[10];`

`float myarray[3284];`

共有十個空間: `c[0]~c[9]`

■ 定義多個同樣型別的陣列

- 格式類似一般的變數
- 範例:

`int b[100], x[50];`

練習

- 請指出下列C程式碼的錯誤為何

```
int  x[8], i;  
for( i = 0; i <= 8; i++)  
    x[i] = i;
```



6.4 使用陣列的例子

■ 初始值

- 使用迴圈給值

```
for(i = 0 ; i < 5; i++)  
    n[i] = 0;
```



常見的程式設計錯誤 6.2

忘了為應該進行初始值設定的陣列元素指定初始值。

- 宣告時就給值

```
int n[5] = {1, 4, 2, 3, 5};  
int n[ ] = {1, 4, 2, 3, 5};  
int n[5] = {0};
```

- 注意：初始值個數過多，會產生語法錯誤

```
int n[2] = {1, 4, 2};
```

- C語言不會自動進行陣列範圍的檢查

```
int n[2];  
for(i=0; i<3; i++)  
    n[i] = 0;
```



```

1  /* Fig. 6.3: fig06_03.c */
3  #include <stdio.h>
6  int main( void )
7  {
8      int n[ 10 ];
9      int i;
10
12     for ( i = 0; i < 10; i++ ) {
13         n[ i ] = 0; /* set element at location i to 0 */
14     }
15
16     printf( "%s%13s\n", "Element", "Value" );
17
19     for ( i = 0; i < 10; i++ ) {
20         printf( "%7d%13d\n", i, n[ i ] );
21     } /* end for */
23     return 0;
24 }

```

for迴圈會分別將陣列中
每個元素初始化

for迴圈會輸出陣列中
每個元素的值

Element	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

課本pp. 6-5

```

1  /* Fig. 6.5: fig06_05.c */
3  #include <stdio.h>
4  #define SIZE 10 /* maximum size of array */
5
7  int main( void )
8  {
10     int s[ SIZE ];
11     int j;
12
13     for ( j = 0; j < SIZE; j++ ) {
14         s[ j ] = 2 + 2 * j;
15     }
16
17     printf( "%s%13s\n", "Element", "Value" );
20     for ( j = 0; j < SIZE; j++ ) {
21         printf( "%7d%13d\n", j, s[ j ] );
22     }
24     return 0;
25 }

```

#define 前置處理器:
用來定義一個常數

編譯器會將SIZE替換成10

for迴圈會分別將陣列中
每個元素初始化

Element	Value
0	2
1	4
2	6
3	8
4	10
5	12
6	14
7	16
8	18
9	20

課本pp. 6-8

```
3  #include <stdio.h>
4  #define SIZE 10 /* maximum size of array */

10  int s[ SIZE ];
13  for ( j = 0; j < SIZE; j++ ) {
14      s[ j ] = 2 + 2 * j;
15  }
```



常見的程式設計錯誤 6.4

在 #define 或 #include 前置處理器之後加上一個分號



常見的程式設計錯誤 6.5

符號常數並不是一個變數。編譯器不會為常數預留記憶體空間



軟體工程的觀點 6.1

請使用符號常數來定義每個陣列的大小，以便讓程式具擴充性



良好的程式設計習慣 6.1

請使用大寫字母來命名符號常數，如此較為醒目

課本pp. 6-9

練習

- 請將陣列中十個數字改用以下兩種方式輸入，再輸出整個陣列。
 - 由使用者鍵盤輸入的方式
 - 由亂數產生的方式



■ 範例：計算陣列中所有元素的總和

```
1  /* Fig. 6.6: fig06_06.c */
3  #include <stdio.h>
4  #define SIZE 12
5
7  int main( void )
8  {
10     int a[ SIZE ] = { 1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45 };
11     int i;
12     int total = 0;
13
15     for ( i = 0; i < SIZE; i++ ) {
16         total += a[ i ];
17     }
18
19     printf( "Total of array element values is %d\n", total );
20     return 0;
21 }
```

Total of array element values is 383

課本pp. 6-9

- 範例：利用陣列來分析某調查所收集的資料
(40位學生對餐廳打分數1~10分，統計分數的分佈)

```
1  /* Fig. 6.7: fig06_07.c */
3  #include <stdio.h>
4  #define RESPONSE_SIZE 40 /* define array sizes */
5  #define FREQUENCY_SIZE 11
6
8  int main( void )
9  {
10     int answer;
11     int rating;
12
14     int frequency[ FREQUENCY_SIZE ] = { 0 };
17     int responses[ RESPONSE_SIZE ] = { 1, 2, 6, 4, 8, 5, 9, 7, 8, 10,
18         1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7, 5, 6, 6,
19         5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };
20
24     for ( answer = 0; answer < RESPONSE_SIZE; answer++ ) {
25         ++frequency[ responses [ answer ] ];
26     }
29     printf( "%s%17s\n", "Rating", "Frequency" );
30
32     for ( rating = 1; rating < FREQUENCY_SIZE; rating++ ) {
33         printf( "%6d%17d\n", rating, frequency[ rating ] );
34     }
36     return 0;
37 }
```

Rating	Frequency
1	2
2	2
3	2
4	2
5	5
6	11
7	5
8	7
9	1
10	3

frequency是一個有11個元素的陣列

responses是存40位學生意見的陣列

responses陣列的內容當作frequency的下標

課本pp. 6-10~11

■ 範例：以長條圖輸出表示陣列元素的內容

```
1  /* Fig. 6.8: fig06_08.c */
3  #include <stdio.h>
4  #define SIZE 10
5
7  int main( void )
8  {
10     int n[ SIZE ] = { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 };
11     int i;
12     int j;
13
14     printf( "%s%13s%17s\n", "Element", "Value", "Histogram" );
17     for ( i = 0; i < SIZE; i++ ) {
18         printf( "%7d%13d", i, n[ i ] );
19
20         for ( j = 1; j <= n[ i ]; j++ ) {
21             printf( "%c", '*' );
22         }
24         printf( "\n" );
25     }
27     return 0;
28 }
```

Element	Value	Histogram
0	19	*****
1	3	***
2	15	*****
3	7	*****
4	11	*****
5	9	*****
6	13	*****
7	5	*****
8	17	*****
9	1	*

使用迴圈控制輸出 * 的個數
第 i 行 印出 n[i] 個星號

課本pp. 6-13

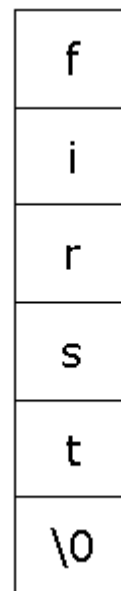
練習

- 請將一顆骰子投擲 n 次，骰子點數由亂數產生數值(1~6)，再輸出1~6點的統計值與長條圖。
 - 利用陣列，且不用switch判斷



■ 使用字元陣列來操作字串

- 陣列可以存放任何型態的資料
- 字串型態的陣列代表一個字串
宣告：
`char arr[6];`
`char arr[] = {'f', 'i', 'r', 's', 't', '\0'};`
`char arr[] = "first";`



注意：

單引號 ' ' 表示字元
雙引號 " " 表示字串

\0 表示結束

- 因此，可以存取個別的字元，`arr[3]` 是 's'

- 輸入：`char arr[6];`
`scanf("%5s", arr);`
輸出：`printf("%s", arr);`

陣列名稱代表陣列在記憶體
的起始位址，因此scanf不需要 &



常見的程式設計錯誤 6.7

提供給 scanf 的字元陣列，不足以放入由鍵盤所讀入的字串時，可能會讓程式漏失資料，或造成其他的執行時期錯誤。

```
1  /* Fig. 6.10: fig06_10.c */
3  #include <stdio.h>
6  int main( void )
7  {
8      char string1[ 20 ];
9      char string2[] = "string literal";
10     int i;
11
12     printf("Enter a string: ");
13     scanf( "%s", string1 );
14
15     printf( "string1 is: %s\nstring2 is: %s\n"
16            "string1 with spaces between characters is:\n",
17            string1, string2 );
18
19     for ( i = 0; string1[ i ] != '\0'; i++ ) {
20         printf( "%c ", string1[ i ] );
21     }
22     printf( "\n" );
23     return 0;
24 }
```

string2 陣列有15個元素(包括 \0)

Enter a string: Hello there
string1 is: Hello
string2 is: string literal
string1 with spaces between characters is:
H e l l o

利用迴圈將陣列中每個元素輸出

課本pp. 6-16

```

1  /* Fig. 6.11: fig06_11.c */
3  #include <stdio.h>
5  void staticArrayInit( void ); /* function prototype */
6  void automaticArrayInit( void );
7
9  int main( void )
10 {
11     printf( "First call to each function:\n" );
12     staticArrayInit();
13     automaticArrayInit();
14
15     printf( "\n\nSecond call to each function:\n" );
16     staticArrayInit();
17     automaticArrayInit();
18     return 0;
19 }

```

First call to each function:

Values on entering staticArrayInit:
array1[0] = 0 array1[1] = 0 array1[2] = 0
Values on exiting staticArrayInit:
array1[0] = 5 array1[1] = 5 array1[2] = 5

Second call to each function:

Values on entering staticArrayInit:
array1[0] = 5 array1[1] = 5 array1[2] = 5
Values on exiting staticArrayInit:
array1[0] = 10 array1[1] = 10 array1[2] = 10

```

22 void staticArrayInit( void )
23 {
24     static int array1[ 3 ];
25     int i;
26     printf( "\nValues on entering staticArrayInit:\n" );
27     for ( i = 0; i <= 2; i++ ) {
28         printf( "array1[ %d ] = %d ", i, array1[ i ] );
29     }
30
31     printf( "\nValues on exiting staticArrayInit:\n" );
32     for ( i = 0; i <= 2; i++ ) {
33         printf( "array1[%d] = %d ", i, array1[i] += 5 );
34     }
35 }

```

static 陣列只會建立與初始化一次

課本pp. 6-17~19

```

44 void automaticArrayInit( void )
45 {
47     int array2[ 3 ] = { 1, 2, 3 };
48     int i;
50     printf( "\n\nValues on entering automaticArrayInit:\n" );
53     for ( i = 0; i <= 2; i++ ) {
54         printf( "array2[ %d ] = %d ", i, array2[ i ] );
55     }
56
57     printf( "\nValues on exiting automaticArrayInit:\n" );
58
60     for ( i = 0; i <= 2; i++ ) {
61         printf( "array2[ %d ] = %d ", i, array2[ i ] += 5 );
62     }
63 }

```

自動變數使得每次呼叫，陣列都會被重新初始化

兩次呼叫的結果
都一樣

First call to each function:

Values on entering automaticArrayInit:

array2[0] = 1 array2[1] = 2 array2[2] = 3

Values on exiting automaticArrayInit:

array2[0] = 6 array2[1] = 7 array2[2] = 8

Second call to each function:

Values on entering automaticArrayInit:

array2[0] = 1 array2[1] = 2 array2[2] = 3

Values on exiting automaticArrayInit:

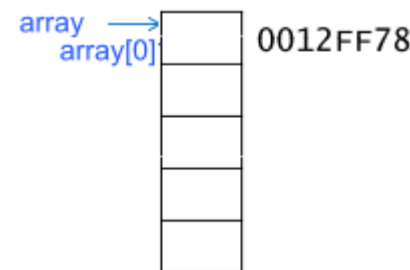
array2[0] = 6 array2[1] = 7 array2[2] = 8

6.5 傳遞陣列給函式

- 若想傳遞陣列給某個函式時，只需指定陣列名稱
 - **陣列名稱**實際上就是此陣列第一個元素的記憶體位址
 - 此傳遞是一種**傳參考**的方式 (非傳值的方式)
 - 以傳參考位址來傳遞陣列，對效率非常有幫助 (省時間、省記憶體)

```
3  #include <stdio.h>
6  int main( void )
7  {
8      char array[ 5 ]; /* define an array of size 5 */
9
10     printf( " array = %p\n&array[0] = %p\n &array = %p\n",
11            array, &array[ 0 ], &array );
12     return 0;
13 }
```

%p 轉換指定詞是用來輸出**記憶體位址**



```
array = 0012FF78
&array[0] = 0012FF78
&array = 0012FF78
```

- 函式參數列需指名希望接收一個陣列

函式原型：

```
void modifyArray(int b[ ], int size); 或是  
void modifyArray(int [ ], int );
```


主程式：

```
int a[5]={0, 1, 2, 3, 4};  
modifyArray(a, 5);
```

函式：

```
void modifyArray(int b[ ], int size){  
    ...  
}
```

告知陣列的元素數量



```

1  /* Fig. 6.13: fig06_13.c */
3  #include <stdio.h>
4  #define SIZE 5
7  void modifyArray( int b[], int size );
8  void modifyElement( int e );
9
11 int main( void )
12 {
13     int a[ SIZE ] = { 0, 1, 2, 3, 4 };
14     int i;
16     printf( "Effects of passing entire array by reference:\n\nThe "
17           "values of the original array are:\n" );
20     for ( i = 0; i < SIZE; i++ ) {
21         printf( "%3d", a[ i ] );
22     }
23
24     printf( "\n" );
27     modifyArray( a, SIZE );
29     printf( "The values of the modified array are:\n" );
32     for ( i = 0; i < SIZE; i++ ) {
33         printf( "%3d", a[ i ] );
34     }
36
37     printf( "\n\nEffects of passing array element "
38           "by value:\n\nThe value of a[3] is %d\n", a[ 3 ] );
40     modifyElement( a[ 3 ] );
41
43     printf( "The value of a[ 3 ] is %d\n", a[ 3 ] );
44     return 0;
45 }
46

```

函式原型指出此函式會接收一個陣列

只傳遞陣列名稱

傳遞 a[3] 的值

課本pp. 6-21~23

```

49 void modifyArray( int b[], int size )
50 {
51     int j;
54     for ( j = 0; j < size; j++ ) {
55         b[ j ] *= 2;
56     }
57 }

```

Effects of passing entire array by reference:

The values of the original array are:

0 1 2 3 4

The values of the modified array are:

0 2 4 6 8

```

61 void modifyElement( int e )
62 {
64     printf( "Value in modifyElement is %d\n", e *= 2 );
65 }

```



Effects of passing array element by value:

The value of a[3] is 6

Value in modifyElement is 12

The value of a[3] is 6

- 特殊的型別修飾詞**const**：用來避免更改陣列的數值

```
1  /* Fig. 6.14: fig06_14.c */
3  #include <stdio.h>
4
5  void tryToModifyArray( const int b[] );
8  int main( void )
9  {
10     int a[] = { 10, 20, 30 };
11
12     tryToModifyArray( a );
13
14     printf("%d %d %d\n", a[0], a[1], a[2]);
15     return 0;
16 }
17
20 void tryToModifyArray( const int b[] )
21 {
22     b[ 0 ] /= 2;
23     b[ 1 ] /= 2;
24     b[ 2 ] /= 2;
25 }
```

const告訴編譯器，陣列不能被修改

Compiling...
FIG06_14.C
fig06_14.c(22): error C2166: l-value specifies const object
fig06_14.c(23): error C2166: l-value specifies const object
fig06_14.c(24): error C2166: l-value specifies const object

練習

- 請問下面程式作些什麼事情？

```
#include <stdio.h>
#define SIZE 10
int whatIsThis( const int b[], int p );

int main( void )
{
    int x;
    int a[ SIZE ] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

    x = whatIsThis( a, SIZE );

    printf( "Result is %d\n", x );
    return 0;
}

int whatIsThis( const int b[], int p )
{
    if ( p == 1 )
        return b[ 0 ];
    else
        return b[ p - 1 ] + whatIsThis( b, p - 1 );
}
```



練習

- 請問下面程式作些什麼事情？

```
#include <stdio.h>
#define SIZE 10
void someFunction( const int b[], int startIndex, int size );

int main( void )
{
    int a[ SIZE ] = { 8, 3, 1, 2, 6, 0, 9, 7, 4, 5 };
    printf( "Answer is:\n" );
    someFunction( a, 0, SIZE );
    printf( "\n" );
    return 0;
}

void someFunction( const int b[], int startIndex, int size )
{
    if ( startIndex < size ) {
        someFunction( b, startIndex + 1, size );
        printf( "%d ", b[ startIndex ] );
    }
}
```



練習

- 請用亂數函式產生n個 0~99 的整數，並將此範圍的整數分成5個等級(0~19, 20~39, 40~59, 60~79, 80~99)。利用陣列方式統計每個等級共產生多少個整數，並利用星號*畫出直方圖。(不用switch)

```
input n: 60

class    value    histogram
  0         15    *****
  1         13    *****
  2         10    *****
  3         14    *****
  4          8    *****

請按任意鍵繼續 . . .
```

