

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

- 定義陣列

 - `int c[12], a[5]={1, 1, 2, 2, 3};`

 - `float myarray[3284]={0};`

- **注意**: 陣列的**下標**開始為 **0**

- 使用迴圈給值

```
int i;
```

```
for(i = 0 ; i < 12; i++)
```

```
    c[i] = 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

6.9 多維陣列

■ 多維陣列

- 具有列(rows)和行(columns)的表格 ($m \times n$ 陣列)
- 類似矩陣：先指定列，再指定行

■ 初始化

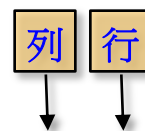
- 以 `int b[3][2] = {{1, 2}, {3, 4}, {5, 6}};`
- 初始值會以列為單位，並用大括號括起來
- 若初始值不夠，剩餘元素會被設為0

`int b[3][2] = {{1}, {3, 4}, {5, 6}};`



1	2
3	4
5	6

1	0
3	4
5	6




■ 引用陣列中的元素

- 指定列，然後指定行 `printf("%d", b[0][1]);`

課本pp. 6-38

	第 0 行	第 1 行	第 2 行	第 3 行
第 0 列	a[0][0]	a[0][1]	a[0][2]	a[0][3]
第 1 列	a[1][0]	a[1][1]	a[1][2]	a[1][3]
第 2 列	a[2][0]	a[2][1]	a[2][2]	a[2][3]



行索引
 列索引
 陣列名稱



常見的程式設計錯誤 6.9

誤以 `a[x, y]` 來引用二維陣列的元素 (應該是 `a[x][y]`)

```

1  /* Fig. 6.21: fig06_21.c */
3  #include <stdio.h>
5  void printArray( const int a[][ 3 ] );
6
8  int main( void )
9  {
11     int array1[ 2 ][ 3 ] = { { 1, 2, 3 }, { 4, 5, 6 } };
12     int array2[ 2 ][ 3 ] = { 1, 2, 3, 4, 5 };
13     int array3[ 2 ][ 3 ] = { { 1, 2 }, { 4 } };
14
15     printf( "Values in array1 by row are:\n" );
16     printArray( array1 );
17
18     printf( "Values in array2 by row are:\n" );
19     printArray( array2 );
20
21     printf( "Values in array3 by row are:\n" );
22     printArray( array3 );
23     return 0;
24 }
27 void printArray( const int a[][ 3 ] )
28 {
29     int i;
30     int j;
33     for ( i = 0; i <= 1; i++ ) {
36         for ( j = 0; j <= 2; j++ ) {
37             printf( "%d ", a[ i ][ j ] );
38         }
40         printf( "\n" );
41     }
42 }

```

完整初始化

部份初始化

```

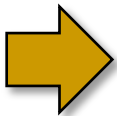
Values in array1 by row are:
1 2 3
4 5 6
Values in array2 by row are:
1 2 3
4 5 0
Values in array3 by row are:
1 2 0
4 0 0

```

課本pp. 6-39

■ 陣列處理都會使用for 重複敘述式

```
a[ 2 ][ 0 ] = 0;  
a[ 2 ][ 1 ] = 0;  
a[ 2 ][ 2 ] = 0;  
a[ 2 ][ 3 ] = 0;
```



```
for(column=0; column<=3; column++){  
    a[2][column] = 0;  
}
```

2	3	1	2
3	4	1	1
0	0	0	0

Q: 如何全部都清成0?

```
for(row = 0; row <= 2; row++){  
    for(column = 0; column <= 3; column++){  
        a[ row ][ column ] = 0;  
    }  
}
```

Q: 如何統計二維陣列所有元素的值？

```
total = 0;
for(row = 0; row <= 2; row++){
    for(column = 0; column <= 3; column++){
        total += a[ row ][ column ];
    }
}
```

- **練習**：模擬投擲兩個骰子**36000**次，統計出現次數。
- **範例**：使用**3*4陣列**儲存三位學生的四次考試成績，程式中使用**函式**來找出
 - **minimum**: 找出所有學生最差的成績
 - **maximum**: 找出所有學生最好的成績
 - **average**: 計算每位學生四科的平均成績

課本pp. 6-41~44


```

1  /* Fig. 6.22: fig06_22.c */
3  #include <stdio.h>
4  #define STUDENTS 3
5  #define EXAMS 4
6
8  int minimum( const int grades[][ EXAMS ], int pupils, int tests );
9  int maximum( const int grades[][ EXAMS ], int pupils, int tests );
10 double average( const int setOfGrades[], int tests );
11 void printArray( const int grades[][ EXAMS ], int pupils, int tests );
12
14 int main( void )
15 {
16     int student;
19     const int studentGrades[ STUDENTS ][ EXAMS ] =
20         { { 77, 68, 86, 73 },
21           { 96, 87, 89, 78 },
22           { 70, 90, 86, 81 } };
25     printf( "The array is:\n" );
26     printArray( studentGrades, STUDENTS, EXAMS );
27
29     printf( "\n\nLowest grade: %d\nHighest grade: %d\n",
30         minimum( studentGrades, STUDENTS, EXAMS ),
31         maximum( studentGrades, STUDENTS, EXAMS ) );
32
34     for ( student = 0; student < STUDENTS; student++ ) {
35         printf( "The average grade for student %d is %.2f\n",
36             student, average( studentGrades[ student ], EXAMS ) );
37     }
39     return 0;
40 }

```

陣列每列對應到一個學生的四科成績

傳遞陣列的某一列給average的函式

課本pp. 6-42

```

43 int minimum( const int grades[][ EXAMS ], int pupils, int tests )
44 {
45     int i;
46     int j;
47     int lowGrade = 100;
48     for ( i = 0; i < pupils; i++ ) {
49         for ( j = 0; j < tests; j++ ) {
50             if ( grades[ i ][ j ] < lowGrade ) {
51                 lowGrade = grades[ i ][ j ];
52             }
53         }
54     }
55     return lowGrade;
56 }

```

```

87 double average( const int setOfGrades[], int tests )
88 {
89     int i;
90     int total = 0;
91     for ( i = 0; i < tests; i++ ) {
92         total += setOfGrades[ i ];
93     }
94     return ( double ) total / tests;
95 }

```

```

101 void printArray( const int grades[][ EXAMS ], int pupils, int tests )
102 {
103     int i;
104     int j;
107     printf( "                [0]  [1]  [2]  [3]" );
110     for ( i = 0; i < pupils; i++ ) {
113         printf( "\nstudentGrades[%d] ", i );
116         for ( j = 0; j < tests; j++ ) {
117             printf( "%-5d", grades[ i ][ j ] );
118         }
119     }
120 }

```

The array is:

	[0]	[1]	[2]	[3]
studentGrades[0]	77	68	86	73
studentGrades[1]	96	87	89	78
studentGrades[2]	70	90	86	81

Lowest grade: 68

Highest grade: 96

The average grade for student 0 is 76.00

The average grade for student 1 is 87.50

The average grade for student 2 is 81.75

6.10 可變長度陣列

■ 可變長度陣列

- 是指陣列的大小是在執行階段時定義的，由使用者來輸入陣列大小，如下程式：

```
int arraySize;  
scanf( "%d", &arraySize );  
int array[ arraySize ];
```

- 不代表執行過程陣列的大小會隨意變動。

課本pp. 6-45

```

3  #include <stdio.h>
6  void print1DArray( int size, int arr[ size ] );
8
9  int main( void )
10 {
11     int arraySize;
13
14     printf( "%s", "Enter size of a one-dimensional array: " );
15     scanf( "%d", &arraySize );
16
24     int array[ arraySize ];
27
29     printf( "\nsizeof(array) yields array size of %d bytes\n",
30         sizeof( array ) );
31
32
33     for ( int i = 0; i < arraySize; ++i ) {
34         array[ i ] = i * i;
35     }
51     puts( "\nOne-dimensional array:" );
52     print1DArray( arraySize, array );
59 } // end main
60
61 void print1DArray( int size, int array[ size ] )
62 {
63
64     for ( int i = 0; i < size; i++ ) {
65         printf( "array[%d] = %d\n", i, array[ i ] );
66     }
67 }

```

回家作業

- 假設某個班級共有學生4人，每人皆修3門課程。請利用一個二維陣列來存放這些隨機產生的成績。
 - 人數和科目請用**define常數**定義
 - 列代表學生; 行代表科目成績
 - 功能:
 - 用表格列印出學生的成績
 - 用函式計算各科的最高分、最低分、平均分及標準差
 - 用函式計算每個人的平均分並輸出第一名的學生



	科目1	科目2	科目3
學生1	92	80	43
學生2	45	65	60
學生3	36	72	40
學生4	31	53	98
學生5	24	26	80

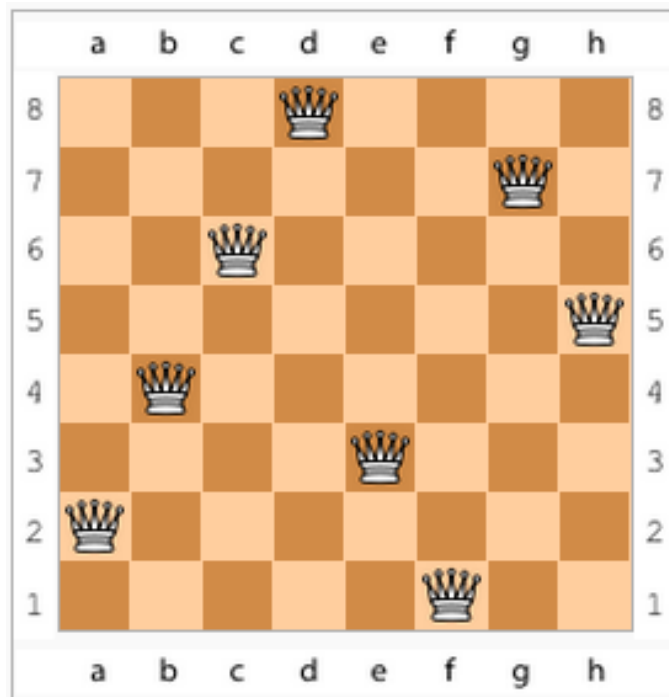
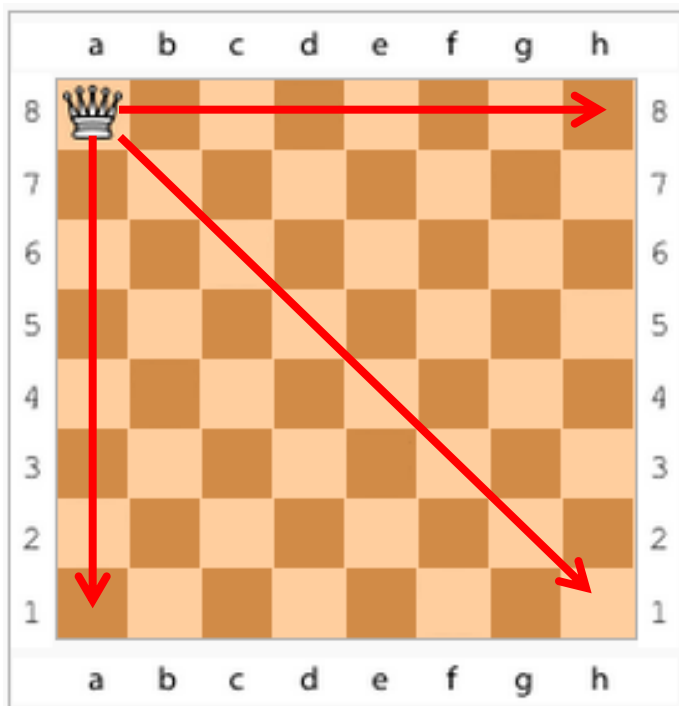
第1科最高分：	92,	最低分：	24,	平均：	45.60,	標準差：	24.19
第2科最高分：	80,	最低分：	26,	平均：	59.20,	標準差：	18.82
第3科最高分：	98,	最低分：	40,	平均：	64.20,	標準差：	22.11

學生1平均分數：	71.67
學生2平均分數：	56.67
學生3平均分數：	49.33
學生4平均分數：	60.67
學生5平均分數：	43.33

第一名為：學生1

八皇后問題(Eight Queens)

- ▶ 如何在8*8西洋棋盤上放置八個皇后，使得任一個皇后都無法直接吃掉其他皇后？注意任兩個皇后不能處在同一行、同一列或是斜對角線上。



課本pp. 6-66
Ex. 6.26, 6.27