

w13-Lab

Two-Dimensional Array

Assembled for 204111
by Ratsameetip Wita

Two-Dimensional Arrays (2D-Arrays)[2]

- เช่น `int val[3][4]`; เป็นการประกาศตัวแปรชื่อ `val` เป็นอาร์เรย์แบบสองมิติ ที่มี 3 แถว และ 4 หลัก

เริ่มนับที่ แถวที่ 0
และหลักที่ 0 เสมอ

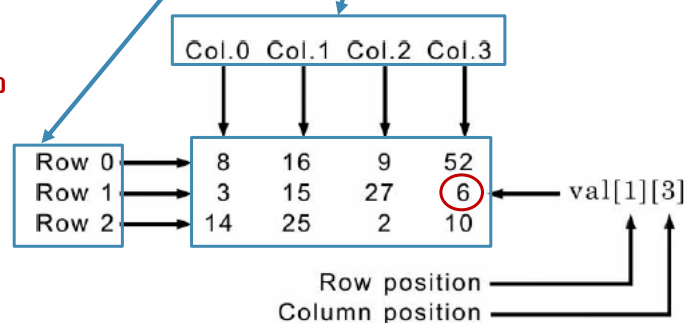


Figure 8.9 Each array element is identified by its row and column

Two-Dimensional Arrays (2D-Arrays)

- 2D Arrays หรืออาร์เรย์แบบสองมิติ จะมีการเก็บข้อมูลในลักษณะคล้ายตาราง
 - มีการอ้างอิงในรูปแบบ แถว และหลัก (Row/Column)
 - ในการประกาศตัวแปร คล้ายกับการประกาศอาร์เรย์ 1 มิติ แต่มีการระบุจำนวน Row และ Column ใน `[][]` ดังนี้
 - `int 2Darray[row][column];`

2D Array Initialization

- Initialization

```
#define NUMROWS 3
#define NUMCOLS 4
int val[NUMROWS][NUMCOLS] = { {8,16,9,52},
                               {3,15,27,6},
                               {14,25,2,10} };
```

ระบุนสมาชิกในอาร์เรย์เรียงลำดับตามแถว ระบุโดย { {แถว 1}, {แถว 2}, {แถว 3} }

- The inner braces can be omitted:

```
int val[NUMROWS][NUMCOLS] =
    {8,16,9,52,3,15,27,6,14,25,2,10};
    val[0][] val[1][] val[2][]
```

- Initialization is done in **row order**

Initialization Order

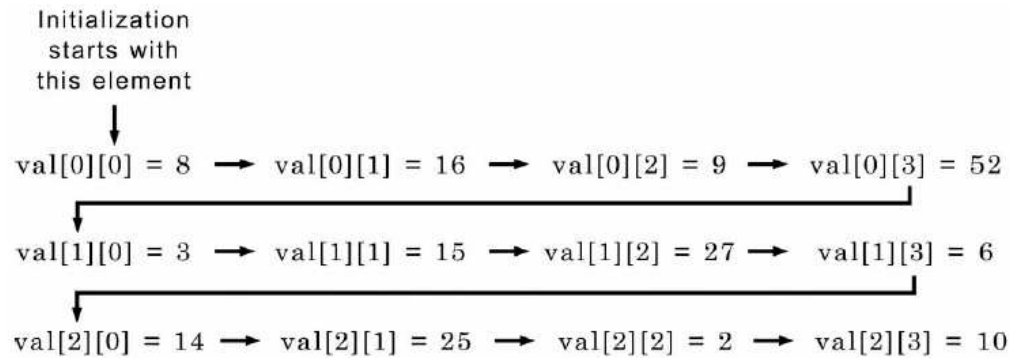


Figure 8.10 Storage and initialization of the val[] array

```
int val[NUMROWS][NUMCOLS] = {8,16,9,52,3,15,27,6,14,25,2,10};
```

```

14 for(i=0;i< NUMROWS;i++){ //วนทีละแถว
15     printf("\n");
16     for(j=0;j<NUMCOLS;j++){ //พิมพ์ทีละหลักในแถวนั้นๆ
17         printf("%2d", val[i][j]);
18     }
19 }
  
```

NUMROWS = 3
NUMCOLS = 4

```

8 16 9 52
3 15 27 6
14 25 2 10
  
```

# of Loop	i	j	val[i][j]
1	0	0	8
2	0	1	16
3	0	2	9
4	0	3	52
5	1	0	3
6	1	1	15
7	1	2	27
8	1	3	6
9	2	0	14
10	2	1	25
11	2	2	2
12	2	3	10

```

01 #include <stdio.h>
02 #define NUMROWS 3
03 #define NUMCOLS 4
04 int main(){
05
06     int val[NUMROWS][NUMCOLS] = {8,16,9,52,3,15,27,6,14,25,2,10};
07     int i, j;
08     printf("Display val array by explicit element\n");
09     printf("%2d %2d %2d %2d\n", val[0][0],val[0][1],val[0][2],val[0][3]);
10     printf("%2d %2d %2d %2d\n", val[1][0],val[1][1],val[1][2],val[1][3]);
11     printf("%2d %2d %2d %2d\n", val[2][0],val[2][1],val[2][2],val[2][3]);
12
13     printf("Display val array using nested loop\n");
14     for(i=0;i< NUMROWS;i++){ //วนทีละแถว
15         printf("\n");
16         for(j=0;j<NUMCOLS;j++){ //พิมพ์ทีละหลักในแถวนั้นๆ
17             printf("%2d", val[i][j]);
18         }
19     }
20     printf("\n");
21     return 0;
22 }
  
```

Display of val array by explicit element

```

8 16 9 52
3 15 27 6
14 25 2 10
  
```

Display of val array using a nested for loop

```

8 16 9 52
3 15 27 6
14 25 2 10
  
```

2D Array Address Ordering

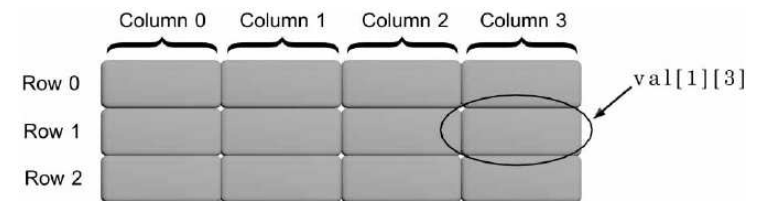
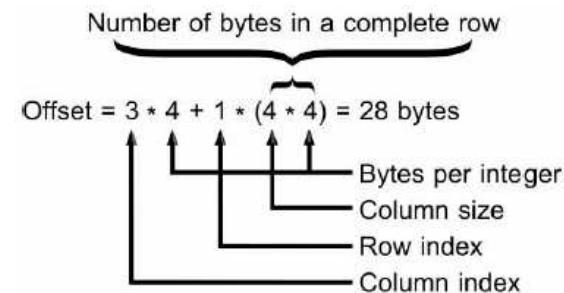


Figure 8.11 Storage of the val array



การหาตำแหน่งของ
element ใน array 2 มิติ

ตัวอย่าง การหาตำแหน่งของ
val[1][3]

Figure 8.12 Determining an element's offset ANSI C, Fourth Edition

n-dimension Array

- ตัวแปรแบบอาร์เรย์ สามารถกำหนดจำนวนมิติ ได้มากกว่า 2 โดยมีการประกาศตัวแปรในรูปแบบดังนี้

```
int 3Darray[10][10][10]; // 3D-Array
double records[100][250][250][250] //4D-Array
```

- โดยในการพิจารณาอาร์เรย์แบบหลายมิติ จะมีการมองข้อมูลเป็นเซตของตาราง

Function Calling with 2D Arrays

- ในการส่งอาร์เรย์สองมิติ เป็นอากิวเมนต์ของฟังก์ชัน จะมีรูปแบบของการใช้งานดังนี้

```
void readarray(int a[ROW][COL], int m, int n);
หรือ
void readarray(int a[][COL], int m, int n);
```

- ROW** และ **COL** คือขนาดของอาร์เรย์ที่จองไว้
- m, n** คือจำนวนสมาชิกที่มีการเก็บข้อมูลจริงในอาร์เรย์

Function Calling with Arrays (Revisited)

- ในภาษา C เราสามารถ ส่งผ่าน array เป็นค่า parameter ไปยัง function ได้ดังนี้

```
void printArray(char s1[],...)
```

```
void printArray(char s1[30],...)
```

```
void printArray(char *s1,...) //204112
```

- ทั้งสามวิธีเป็นการส่ง array ไปยัง function ในรูปแบบการส่ง memory address ทั้งสิ้น

Practice I

- ให้เขียนโปรแกรมเพื่อรับ Matrix ขนาด $m \times n$ 2 ชุด ($1 \leq m \leq 20$) ($1 \leq n \leq 20$) และแสดงผลบวกของ Matrix โดยรับค่า Input ผ่าน IO Redirection และแสดงผลลัพธ์ที่หน้าจอ

	data.txt	Output
row	3	11 13 15 17
col	4	19 21 23 25
matrix 1	1 2 3 4	27 29 31 33
	5 6 7 8	
	9 10 11 12	
matrix 2	10 11 12 13	
	14 15 16 17	
	18 19 20 21	

Practice II

- ให้เขียนโปรแกรมเพื่อแสดง transpose ของ Matrix ขนาด $m \times n$ ($1 \leq m \leq 20$) ($1 \leq n \leq 20$) โดยรับค่า Input ผ่าน IO Redirection และแสดงผลที่หน้าจอ

```

data.txt
row 3
col 4
matrix 1 {
1 2 3 4
5 6 7 8
9 10 11 12
}

Output
1 5 9
2 6 10
3 7 11
4 8 12

```

Practice III : Matrix dot product

- ให้เขียนโปรแกรมเพื่อรับค่า Matrix ขนาด $m \times n$ และ $n \times o$ และทำการคำนวณและแสดงผลการคูณแบบ dot product ของ Matrix โดยรับค่า Input ผ่าน IO Redirection และแสดงผลที่หน้าจอ

```

data.txt
row col of A 3 2
row col of B 2 3
matrix A {
1 2 3
4 5 6
7 8
}
matrix B {
9 10
11 12
}

```

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 \\ 64 \end{bmatrix}$$

"Dot Product"

```

Output
58 64
139 154

```