

ตัวแปรชุด (Array)

CompSci CMU

ตัวอย่าง pseudo code

กำหนดค่าเริ่มต้นให้กับสมาชิกทุกตัวของ score มีค่าเป็น 10

```
Let I ← 0
Repeat
    Set 10 to score[I]
    Add 1 to I
Until I > 9
```

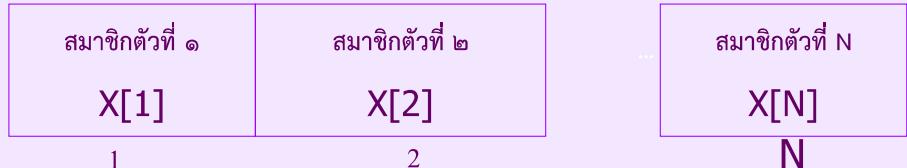
10	10	...	10
score[0]	score[1]		score[9]

cs112

3

ตัวแปรชุด

- กลุ่มของข้อมูลซึ่งมีชนิดข้อมูลเป็นอย่างเดียวกัน อ้างถึงกลุ่มข้อมูลชุดนี้ด้วยชื่อตัวแปรเดียวกัน
- ข้อมูลแต่ละตัวเรียกว่าเป็นสมาชิกของตัวแปรชุด
- สมาชิกแต่ละตัวจะมีหมายเลข/ดัชนี(index)กำกับ
- อ้างถึงสมาชิกแต่ละตัวด้วยชื่อตัวแปรชุดพร้อมระบุหมายเลข/ดัชนี



cs112

2

ตัวแปรชุด

✿ การประกาศตัวแปรชุด

✿ การอ้างถึงหรือการเรียกใช้ข้อมูลของตัวแปรชุด

✿ ความสัมพันธ์ระหว่างตัวแปรชุดกับตัวแปรพอยน์เตอร์

✿ การส่งผ่านตัวแปรชุดให้กับฟังก์ชัน

cs112

4



การประกาศตัวแปรชุด

- รูปแบบ (1) ชนิดข้อมูล ชื่อตัวแปร [จำนวนสมาชิก];

```
int score[10];
```

```
char id[8], name[N];
```

N เป็นค่าคงที่สัญลักษณ์ กำหนดโดย #define N 25

- การจัดสรรเนื้อที่ในหน่วยความจำ

สมาชิกแต่ละตัวจะถูกจัดสรรที่ในหน่วยความจำแบบต่อเนื่อง
ขนาดหน่วยความจำของแต่ละตัวจะขึ้นกับชนิดข้อมูล
เช่นตัวแปร score จะใช้เนื้อที่ขนาด $4 \times 10 = 20$ ไบท์

สมาชิกตัวที่ 1	สมาชิกตัวที่ 2	...	สมาชิกตัวที่ 10
4 ไบท์	4 ไบท์	4 ไบท์	4 ไบท์

id จะใช้เนื้อที่ขนาด $1 \times 8 = 8$ ไบท์

name จะใช้เนื้อที่ขนาด $1 \times 25 = 25$ ไบท์

c5112

5



ตัวอย่างชุดคำสั่งภาษา C

- กำหนดค่าเริ่มต้นให้กับสมาชิกทุกตัวของ score มีค่าเป็น 0

```
int i, score[10];
```

```
for (i= 0; i<10; i++)
```

```
score[i] = 0;
```

0	0	...	0
score[0]	score[1]		score[9]

c5112

7

การอ้างถึงหรือการเรียกใช้ข้อมูลของตัวแปรชุด

- รูปแบบ ชื่อตัวแปรชุด[ตัวนี้ หรือ ตัวบ่งชี้]

- ในภาษา C ดังนี้

- มีชนิดข้อมูลเป็นจำนวนเต็ม

- ดัชนีตัวแปรมีค่าเป็น 0 เสมอ

- อาจอยู่ในรูปค่าคงที่ หรือ ตัวแปร หรือ นิพจน์ที่ให้ค่าเป็นจำนวนเต็ม

score	สมาชิกตัวที่ 1	สมาชิกตัวที่ 2	...	สมาชิกตัวที่ 10
-------	----------------	----------------	-----	-----------------

ตัวบ่งชี้	0	1	...	9
-----------	---	---	-----	---

score [0] /* อ้างถึงสมาชิกตัวแรกของ score */

score [i] /* สมาชิกที่อยู่ดัดจากตัวแรกไป i ตัว */

score [i+1] /* สมาชิกที่อยู่ดัดจากตัวแรกไป i+1 */

c5112

6



การประกาศตัวแปรชุดพร้อมกับกำหนดค่าเริ่มต้น

- รูปแบบ

(2) ชนิดข้อมูล ชื่อตัวแปร [] = {ค่าเริ่มต้นตัวที่ 1, ค่าเริ่มต้นตัวที่ 2, ค่าเริ่มต้นตัวสุดท้าย};

(3) ชนิดข้อมูล ชื่อตัวแปร [n] =

{ค่าเริ่มต้นตัวที่ 1, ค่าเริ่มต้นตัวที่ 2,..., ค่าเริ่มต้นตัวสุดท้าย};
n ค่า

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

5	2	4	1	3
X[0]	X[1]	X[2]	X[3]	X[4]

char id[8] = { '2','3','0','5','1','7','1','\0' };

int X[5] = {};

'2'	'3'	'0'	'5'	'1'	'7'	'1'	'\0'
-----	-----	-----	-----	-----	-----	-----	------

id[0] id[1] id[2] id[3] id[4] id[5] id[6] id[7]

0	0	0	0	0
X[0]	X[1]	X[2]	X[3]	X[4]

8



ตัวอย่างชุดคำสั่งภาษา C



เรียนรู้ดับคณ์แบบน้อยไปมาก

```
#define N 5
:
int i, j, tmp, score[N] = {5, 2, 4, 1, 3};

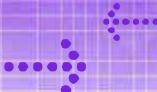
for (i = 0; i < N - 1; i++)
    for (j = i + 1; j < N; j++)
        if (score[i] > score[j]) {
            tmp = score[i];
            score[i] = score[j];
            score[j] = tmp;
        }
```

cs112

9



ตัวอย่าง



จงเขียนโปรแกรมเพื่อหาคณ์แบบเฉลี่ยของผลสอบกลางภาคของนักศึกษาจำนวน 10 คน และแสดงคณ์แบบที่นักศึกษาแต่ละคนได้รับ พร้อมทั้งทำเครื่องหมาย '*' หลังคณ์แบบสูงสุด และทำเครื่องหมาย '-' หลังคณ์แบบที่มีค่าต่ำกว่าคณ์แบบเฉลี่ย

cs112

11

แบบฝึกหัด



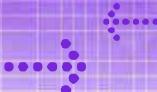
- จงเขียนโปรแกรมรับ user name ซึ่งมีความยาว 4-8 ตัวอักษร และทำการเข้ารหัสด้วยสูตร รหัสใหม่ = รหัสเดิม + 26

cs112

10



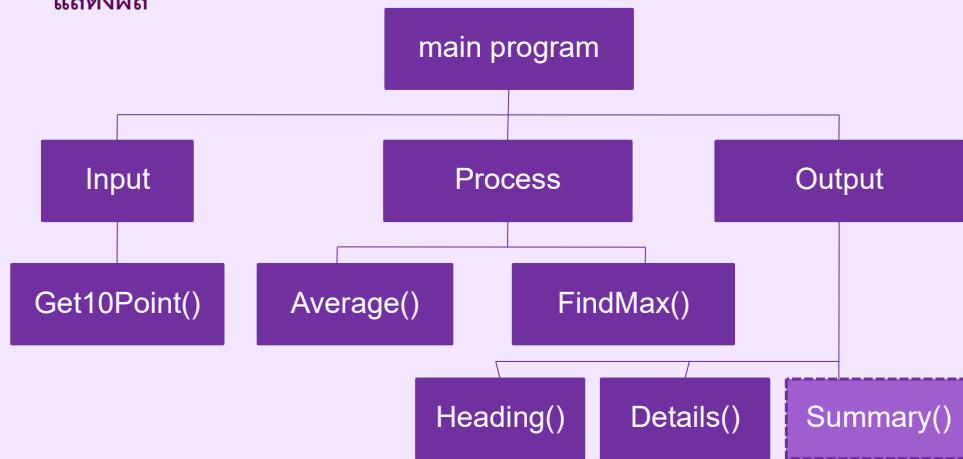
วิเคราะห์



- Input: mark 10 values (X)
- Output: Average, Maximum and list of all 10 marks displayed with '*' if it's maximum, or '-' for all values are less than Average
- Process
 - Input data
 - Find maximum and average
 - Print list of all data with condition marking
- การใช้ตัวแปร จำเป็นต้องให้ X เป็นตัวแปรชุด
 - เนื่องจาก คณ์แบบ 10 ค่าที่ต้องนำเข้าນั้น ถูกใช้ในการประมวลผลหลายรอบ ได้แก่ ใช้ใน การหาค่าเฉลี่ย หาค่าสูงสุด และใช้ในการแสดงผล
 - เพื่อลดโอกาสของการเกิดข้อผิดพลาด(เนื่องจากมนุษย์)นำเข้าข้อมูล จึงควรนำเข้ารอบ เดียวแต่เก็บค่าที่รับได้ไว้สำหรับการประมวลผล ต่อไป

ออกแบบโปรแกรม โดยใช้หลักการ Top-down

- โครงสร้างของโปรแกรม แบ่งเป็น 3 ส่วน ได้แก่ รับข้อมูล ประมวลผล และแสดงผล



204112

13

```
#include <stdio.h>
#define N 10
int main( ) {
    float X[N], mean, max, sum;
    int i;
    // get 10 points into X
    for (i=0; i < N; i++) {
        printf("Enter number : ");
        scanf("%.2f", &X[i]);
    }
    // Find average of 10 points
    for (i=0, sum=0.0; i<N; i++)
        sum += X[i];
    mean = sum/N;
    // Find the maximum point
    for (i=1, max=X[0]; i<N; i++)
        if (max<X[i])
            max=X[i];
    // Display 10 points with marker
    for (i=0; i<N; i++) {
        printf("X[%d] is %.2f ", i+1, X[i]);
        if (X[i]==max)
            putchar('*');
        if (X[i] < mean)
            putchar('-');
        putchar('\n');
    }
    return 0;
}
```

cs112

14

ตัวแปรชุด

การประกาศตัวแปรชุด

การอ้างถึงหรือการเรียกใช้ข้อมูลของตัวแปรชุด

ความสัมพันธ์ระหว่างตัวแปรชุดกับตัวแปรพ้อยน์เตอร์

การส่งผ่านตัวแปรชุดให้กับฟังก์ชัน

cs112

15

ความสัมพันธ์ระหว่างตัวแปรชุดกับตัวแปรพ้อยน์เตอร์

- ตัวแปรพ้อยน์เตอร์ เป็น
 - ตัวแปรซึ่งเก็บค่าแอดเดรสหรือตำแหน่งในหน่วยความจำ
 - เราสามารถเปลี่ยนแปลงค่าของตัวแปรพ้อยน์เตอร์ได้
- ตัวแปรชุด เป็นชื่อที่เราใช้อ้างถึงกลุ่มของข้อมูล ในภาษา C เมื่ออ้างถึงชื่อตัวแปรชุดเราจะได้เป็นค่าของตำแหน่งใน หน่วยความจำ เราไม่สามารถเปลี่ยนค่านี้ได้

```
int x[ ] = {10, 20, 30, 40, 50};
```

10	20	30	40	50
x[0]	x[1]	x[2]	x[3]	x[4]

เมื่ออ้างถึง x เราจะได้ แอดเดรสของสมาชิกตัวแรก ดังนั้น

*x *(x+1) *(x+2) *(x+3) *(x+4)

16



ตัวอย่างชุดคำสั่ง (1)



```
/* Demo array and address of each element*/
#include <stdio.h>
void main (void) {
    int data[ ] = {10, 20, 30, 40, 50}, index;

    printf("Starting address of array is %u\n", data);
    for (index=0; index < 5; index++)
        printf("address of element %u is %u, data is %d\n",
               index + 1, data + index, *(data + index));
}
```

cs112

17

ตัวอย่างชุดคำสั่ง (2)

```
/* การหาผลรวมของสมาชิกในตัวแปรชุด โดยใช้ดัชนี*/
for (i=0, sum=0; i<N; ++i)
    sum += A[i];
/* โดยใช้ตัวแปรอยู่เตอร์ */
for (p=A; p <&A[N]; p++)
    sum += *p;

/* โดยใช้ตัวแปรชุดสมมูลเป็นพอยน์เตอร์*/
for (i=0, sum=0; i<N; ++i)
    sum += *(A + i);
/* โดยใช้ตัวแปรพอยน์เตอร์สมมูลเป็นตัวแปรชุด*/
for (p=A, i=0, sum=0; i<N; ++i)
    sum += p[i];
```

cs112

18



การส่งผ่านตัวแปรชุดให้กับฟังก์ชัน



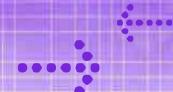
- Call by value :** ส่งได้แค่สมาชิกของตัวแปรชุด
 - ที่ส่วนของการประกาศพารามิเตอร์ ประกาศให้พารามิเตอร์มีชนิดข้อมูลเป็นอย่างเดียวกับค่าที่ต้องการส่งผ่าน
- Call by reference :** ส่งเป็น address ของสมาชิกตัวแรกของตัวแปรชุดที่ต้องการประมวลผล
 - ที่ส่วนหัวของการประกาศฟังก์ชัน (Function header) ในส่วนของการประกาศพารามิเตอร์ หากพารามิเตอร์เป็นตัวแปรชุด 1 มิติไม่ต้องระบุจำนวนสมาชิก แต่ถ้าเป็นตัวแปรชุด 2 มิติ ให้ระบุเฉพาะจำนวนสมาชิกของมิติที่สอง (จำนวนคอลัมน์)
 - พารามิเตอร์ที่ใช้รับค่าของตัวแปรชุด อาจจะถูกประกาศในลักษณะของ ตัวแปรพอยน์เตอร์ได้

cs112

19



การส่งผ่านตัวแปรชุดให้กับฟังก์ชัน



```
// Function to sum elements of an array
// array A is passed as a pointer, Size of array is passed as separate int
int sum (int A[], int Size) {
    int i, s=0;
    for (i=0; i<Size; i++)
        s += A[i];
    return s;
}
// at the calling side
int sum ( int [], int ) ; // prototype of sum function

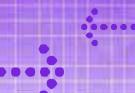
// Various ways that sum() might be called
Invocation
sum(v, 100)
sum(&v[7], k-7)
sum(v+7, 2*k)
What gets computed and returned
```

cs112

20



ตัวอย่าง(เดิม)



จงเขียนโปรแกรมเพื่อหาคะแนนเฉลี่ยของผลสอบกลางภาคของนักศึกษาจำนวน 10 คน แล้วแสดงคะแนนที่นักศึกษาแต่ละคนได้รับ พร้อมทั้งทำเครื่องหมาย '*' หากคะแนนสูงสุด และทำเครื่องหมาย '-' หากคะแนนที่มีค่าต่ำกว่าคะแนนเฉลี่ย

- Input: mark 10 values (X)
- Output: Average, Maximum and list of all 10 marks displayed with '*' if it's maximum, or '-' for all values are less than Average
- Process
 - Input data
 - Find maximum and average
 - Print list of all data with condition marking
- การใช้ตัวแปร จำเป็นต้องให้ X เป็นตัวแปรชุด
 - เนื่องจาก คะแนน 10 ค่าที่ต้องนำเข้านั้น ถูกใช้ในการประมวลผลหลายรอบ ได้แก่ ใช้ในการหาค่าเฉลี่ย หาค่าสูงสุด และใช้ในการแสดงผล
 - เพื่อลดโอกาสของการเกิดข้อผิดพลาด(เนื่องจากมนุษย์)นำเข้าข้อมูล จึงควรนำเข้ารอบเดียวแต่เก็บค่าที่รับได้ ไว้สำหรับการประมวลผล ต่อไป

21



```
#include <stdio.h>
#define N 10
void Get10Point(float [], int);
float Average(float [], int);
float MaxPoint(float [], int);
void Heading( void );
void Details(float [], int, float, float);
int main() {
    float X[N];
    float Mean;

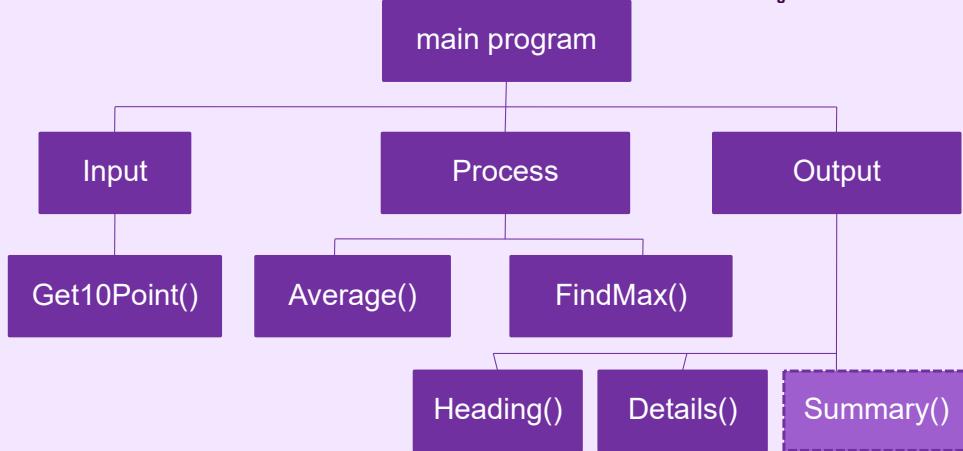
    Get10Point(X, N);
    Mean = Average(X, N);
    Heading();
    Details(X, N, Mean, MaxPoint(X, N));
    return 0;
}
```

23



ออกแบบโปรแกรม โดยใช้หลักการ Top-down

- โครงสร้างของโปรแกรม แบ่งเป็น 3 ส่วน ได้แก่ รับข้อมูล ประมวลผล และแสดงผล
- ทำการออกแบบให้ละเอียดขึ้นโดยแบ่งแต่ละการประมวลผล เป็น模塊



204112

22



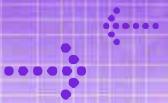
```
void Get10Point(float y[ ], int n)
{
    int i;
    for (i=0; i < n; i++) {
        printf("Enter number : ");
        scanf("%2f", &y[i]);
    }
}

float Average (float y[], int n) {
    int i; float s;
    for (i=0, s=0; i<n; i++)
        s += y[i];
    return s/n;
}

void Details (int y[], int n,
              float mean, float max)
{
    int i;
    for (i=0; i<n; i++) {
        printf("X[%d] is %.2f ", i+1, y[i]);
        if (y[i]==max)
            putchar('*');
        if (y[i] < mean)
            putchar('-');
        putchar('\n');
    }
}
```

204112

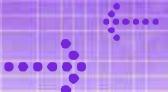
24



ตัวแปรชุดของอักขระ String



เนื้อหา



- การใช้ฟังก์ชัน gets() และ puts()
- ฟังก์ชันสำหรับการเชื่อมต่อข้อความ strcat()
- ฟังก์ชันเปรียบเทียบข้อความ 2 ชุด strcmp()
- ฟังก์ชันคัดลอกข้อความหนึ่งจากข้อความอีกชุด strcpy()
- ฟังก์ชันเพื่อทำการนับความยาวของข้อความ strlen()

String (27)



ตัวแปรชุดของอักขระ(String)

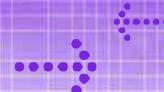


- กลุ่มของข้อมูลซึ่งมีชนิดข้อมูลเป็นอักขระ บางครั้งจะถูกเรียกว่า สายอักขระ หรือ ข้อความ
- การจัดการกับตัวแปรลักษณะนี้ ทำเช่นเดียวกับการจัดการกับตัวแปรชุด 1 มิติทั่วไป
- แต่เนื่องจากลักษณะหรือการจัดการกับสายอักขระของงานประยุกต์เป็นลักษณะเดียวกัน ทางผู้ผลิตตัวแปรภาษาจึงมีฟังก์ชัน สำเร็จรูป เพื่อจัดการกับสายอักขระไว้พร้อมในไลบรารีให้เรียกใช้ได้

String (26)



การใช้ฟังก์ชัน gets()



- gets() เพื่อรับข้อความ 1 บรรทัด (จบด้วยการเคาะ Enter) จากคีย์บอร์ดซึ่งเป็นอุปกรณ์รับเข้ามาตรฐาน (standard input device, stdin)
- Include file : <stdio.h>
- Prototype : char * gets (char *);
- Arguments : Storage location for input string
- Returns : A pointer to its argument if successful,
a NULL pointer if at end-of-file or unsuccessful

String (28)



ตัวอย่างการใช้ฟังก์ชัน gets()

char s[10];

s	H	e			o	\0	O	O	\0	
0	1	2	3	4	5	6	7	8	9	

gets(s);

H e | | o <Enter>

gets(s+5);

OOO<Enter>

String (29)



ตัวอย่างการใช้ฟังก์ชัน puts()

s	H	e			o	\0	\0	\0	\0	
0	1	2	3	4	5	6	7	8	9	

puts(s);

Hello\000

puts(s+3);

lo\000

puts("Hi!");

Hi!

String (31)



การใช้ฟังก์ชัน puts()

- puts() เพื่อการแสดงข้อความ โดยจะนำอักขระที่ไม่ใช่ null character ซึ่งอยู่ในตำแหน่งที่ระบุ มาแสดง เมื่อพบอักขระ null จะแสดงเป็นอักขระ \n นั่นคือการขึ้นบรรทัดใหม่
- Include file : <stdio.h>
- Prototype : int puts (const char *);
- Arguments : string to be output
- Returns : 0 if successful, nonzero if not

String (30)



การใช้ฟังก์ชัน strcat()

- char * strcat (char * s1, const char *s2);
- strcat() นำข้อความจากสตริง s2 ไปต่อท้ายสตริง s1 นั่นคือ นำ อักขระแรกที่ไม่ใช่ null character ของสตริง s2 ไปวางลงใน ตำแหน่ง null character ของสตริง s1

s1 H i \0

s2 ! \0

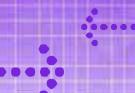
strcat(s1, s2);

s1 H i ! \0

String (32)



การใช้ฟังก์ชัน strcmp()



- int strcmp (const char *s1, const char *s2);
- Compares two strings lexicographically
 - if s1 = s2, returns 0
 - if s1 < s2, returns -1
 - if s1 > s2, returns 1

s1 H i \0

s2 ! \0

R = strcmp(s1, s2);

→ 1

R = strcmp(s1, "Ho");

→ -1

String (33)



การใช้ฟังก์ชัน strcpy()



- char *strcpy (char *s1, const char *s2);
- Copies s2 to s1

s1 H i \0

s2 O H \0

strcpy(s1, s2);

s1 O H \0

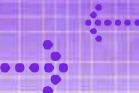
strcpy(s2, "Coucou");

s2 C o u c o u \0

String (34)



แบบฝึกความเข้าใจการใช้ฟังก์ชัน



Declarations and initializations

char s1[] = "beautiful big sky country",

s2[] = "how now brown cow";

Expression	Value
------------	-------

strlen(s1)	
------------	--

strlen(s2 + 8)	
----------------	--

strcmp(s1, s2)	
----------------	--

Statements	What gets printed
------------	-------------------

puts(s1+10);	
--------------	--

strcpy(s1+10, s2+8);	
----------------------	--

strcat(s1, "s!");	
-------------------	--

puts(s1);	
-----------	--



```
#include <stdio.h>
```

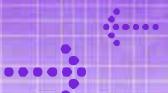
```
#define Size 128
```

```
int main()
```

```
    int i;
    char s[Size];
```

```
    puts("Enter any string with length  
not more than 127 character");
    gets(s);
    puts("Input string is ...");
```

```
    puts(s);
    return 0;
}
```



ตัวแปรชุดหลายมิติ

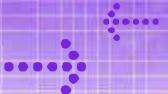
- การประกาศ
- การจัดสรรพื้นที่หน่วยความจำ
- การเข้าถึงสมาชิก
 - ยึดแคล เป็นหลัก (Row major)
 - ยึดสุดมร เป็นหลัก (Column major)
- การส่งผ่านตัวแปรชุดให้กับฟังก์ชัน

204112

38



การประกาศตัวแปรชุด 2 มิติ (Two Dimensions)



- รูปแบบ
 - ชนิดข้อมูลของสมาชิก ชื่อตัวแปร $[n][m]$;
 - ชนิดข้อมูลของสมาชิก ชื่อตัวแปร $[n][m] = \{c_{11}, c_{12}, \dots, c_{1m}\}, \dots, \{c_{n1}, c_{n2}, \dots, c_{nm}\}$;
 - ชนิดข้อมูลของสมาชิก ชื่อตัวแปร $[][] = \{c_{11}, c_{12}, \dots, c_{1m}\}, \dots, \{c_{n1}, c_{n2}, \dots, c_{nm}\}$;

เมื่อ n หมายถึง จำนวนแคล และ m หมายถึง จำนวนสุดมร
จำนวนสมาชิกของตัวแปรชุดนี้ มี $n \times m$ ตัว

- ตัวอย่าง

```
int table[2][3];
char answer[2][4]= {'Y', 'e', 's', '\0'},
                  {'N', 'o', '\0', {}}; /*{"Yes", "No"}*/
float stu[][]={{2.0, 3.5}, {1.85, 3.75}, {4.0, 4.0}};
```

39

ตัวแปรชุด 2 มิติ (two dimensional arrays)

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

	สุดมร	0	1	2
แคล	0	a[0][0]	a[0][1]	a[0][2]
	1	a[1][0]	a[1][1]	a[1][2]

cs112

40



การจัดสรรพื้นที่หน่วยความจำ

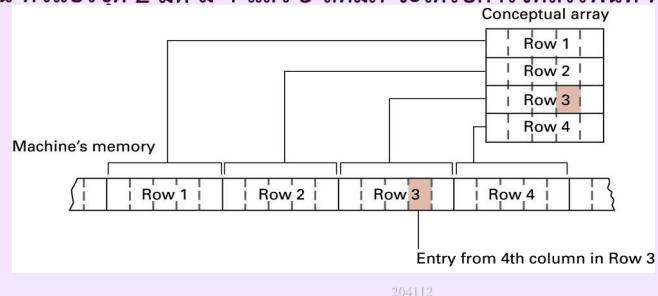
✿ จัดสรรพื้นที่ที่ต่อเนื่อง ตามลำดับแบบแอกาเป็นหลัก (**Row major order**)

✿ ขนาดพื้นที่ที่ถูกจัดสรร เท่ากับ $\prod_i^n S_i \times B$

เมื่อ S_i คือขนาดของมิติที่ i n แทนจำนวนมิติทั้งหมด

B ขนาดของหน่วยความจำต่อหนึ่งหน่วยข้อมูล เช่น เลขจำนวนเต็ม **B** จะมีค่าเป็น 2 ไบต์

✿ เช่น ตัวแปรชุด 2 มิติ มี 4 แถว 5 สดมก จะได้รับการจัดสรรพื้นที่ ดังรูป



204112

41



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

สดมก	0	1	2
แถว 0	<code>a[0][0]</code>	<code>a[0][1]</code>	<code>a[0][2]</code>
แถว 1	<code>a[1][0]</code>	<code>a[1][1]</code>	<code>a[1][2]</code>

สดมก	0	1	2
แถว 0	1	2	3
แถว 1	4	5	6



c5112

43



ตัวแปรชุดหลายมิติ

✿ ตัวแปรชุด 2 มิติ (two dimensional arrays)

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

แถว 0	1	2	3
แถว 1	4	5	6

✿ เพื่อเข้าให้ถึงข้อมูลซึ่งเป็นสมาชิกของตัวแปรชุด จะต้องใช้ตัวบ่งชี้เท่ากับจำนวนมิติ
`a[0][0]` /*สมาชิกที่อยู่ในแถวแรก สดมกแรก*/
`a[1][2]` /*สมาชิกที่อยู่ในแถวที่สอง สดมกที่ ๓ */
`a[k][k+1]` /*สมาชิกที่อยู่ในแถว k สดมก k+1*/



42



การอ้างถึงสมาชิก

- จำนวนด้านนี้ที่จำเป็นต้องใช้เพื่อรับถึงสมาชิกของตัวแปรชุด ต้องเท่ากับจำนวนมิติของตัวแปรชุดนั้น
- เช่น ตัวแปรชุด 2 มิติ

`a[i][j]` /*สมาชิกที่อยู่ในแถว i สดมก j*/

สดมก	0	1	2
แถว 0	<code>a[0][0]</code>	<code>a[0][1]</code>	<code>a[0][2]</code>
แถว 1	<code>a[1][0]</code>	<code>a[1][1]</code>	<code>a[1][2]</code>

204112

44



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

1	2	3	4	5	6
---	---	---	---	---	---

a : address of the first row

*a : address of the 1st column of the 1st row

**a : element of the 1st column of the 1st row

a[0][0] /*สมาชิกที่อยู่ในแควรร์ สมมติแก้*/

*a + j : address of the jth column of the 1st row

* (a + i) : address of the 1st column of the ith row

* (a + i)+j : address of the jth column of the ith row

((a + i)+j) : element of the jth column of the ith row

*(a[i] + j) = ? *(&a[0][0] + 3*i + j) = ?

CS112

45



การเข้าถึงสมาชิก

- ยึดแควร์เป็นหลัก (Row major order)

สมมติ	0	1	2
แควร์ 0	a[0][0]	a[0][1]	a[0][2]
แควร์ 1	a[1][0]	a[1][1]	a[1][2]

- ยึดสมมติเป็นหลัก (Column major order)

สมมติ	0	1	2
แควร์ 0	a[0][0]	a[0][1]	a[0][2]
แควร์ 1	a[1][0]	a[1][1]	a[1][2]

204112

47



ตัวอย่างทดลอง

```
#include <stdio.h>
int main() {
    char      p[2][3][4] = {'a', 'b', 'c', 'd', 'e', 'f',
                           'g', 'h', 'i', 'j', 'k', 'l',
                           'm', 'n', 'o', 'p', 'q', 'r',
                           's', 't', 'u', 'v', 'w', 'x'};
    int      i, j, k;
    printf("p=%x, *p=%x, **p=%x, ***p=%x\n",
           p, *p, **p, ***p);
    for (i=0; i<2; i++)
        for (j=0; j<3; j++)
            for (k=0; k<4; k++)
                printf("p[%d][%d][%d] = %c\t", i, j, k,
                       p[i][j][k]);
    printf("\n");
    return 0;
}
```

204112

46



ตัวอย่าง ยึดแควร์เป็นหลัก (Row major order)

- เขียนชุดคำสั่งภาษาซีปรับปรุงค่าสมาชิกของตัวแปรชุด a โดยให้ค่าใหม่เป็นลิบเท่าของค่าเดิม

สมมติ	0	1	2
แควร์ 0	10	20	30
แควร์ 1	40	50	60

```
#define ROW 2
#define COL 3
int i, j, a[][]={{1, 2, 3},{4, 5, 6}};

for (i=0; i<ROW; i++)
    for (j=0; j<COL; j++)
        a[i][j] = a[i][j]*10; /* a[i][j] *= 10; */
```

48



ตัวอย่าง ยืดสมมติเป็นหลัก (Column major order) ⏮

- เขียนชุดคำสั่งภาษาซีปรับปรุงค่าสมาชิกของตัวแปรชุด a โดยให้ค่าใหม่เป็นลิบเท่าของค่าเดิม

สมมติ	0	1	2
แถว 0	10	20	30
แถว 1	40	50	60

```
#define ROW 2
#define COL 3
int i, j, a[][]={{1, 2, 3}, {4, 5, 6}};

for (j=0; j<COL; j++)
    for (i=0; i<ROW; i++)
        a[i][j] = a[i][j]*10; /* a[i][j] *= 10; */
```



ตัวอย่าง

ชุดคำสั่งเพื่อสร้างเมทริกเอกลักษณ์ (Identity matrix) ขนาด $n \times n$

Identity matrix เป็นเมทริกซ์ที่ค่าสมาชิกทุกตัวมีค่าเป็น 0 ยกเว้นสมาชิกในแนวตระแยงที่ตำแหน่งแถวเท่ากับตำแหน่งคอลัมน์จะมีค่าเป็น 1

เช่น : เป็นเมทริกซ์เอกลักษณ์ขนาด 3×3

```
#define Size 3
:
int m, n, I [Size][Size];
I = 
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

for (m = 0; m < Size; m++)
    for (n = 0; n < Size; n++)
        I [m][n] = (m == n ? 1 : 0);
```



การออกแบบขั้นตอนวิธีที่สอดรับกับรูปแบบการจัดการหน่วยจำของคอมพิวเตอร์ (โดยตัวแปลภาษาหนึ่งหรือระบบปฏิบัติการ) ช่วยให้สมรรถนะของโปรแกรม (Program Performance) ดีขึ้นได้มากๆ ดังนั้นการเขียนโปรแกรมภาษาซีจัดการตัวแปรชุดหลาย มิติจึงควรใช้หลักการ Row major order



การส่งผ่านตัวแปรชุดให้กับฟังก์ชัน

- Called by value : ส่งได้แค่สมาชิกของตัวแปรชุด
 - ที่ส่วนของการประกาศพารามิเตอร์ ประกาศให้พารามิเตอร์มีชนิดข้อมูลเป็นอย่างเดียวกับค่าที่ต้องการส่งผ่าน
- Called by reference : ส่งเป็น address ของสมาชิกตัวแรกของตัวแปรชุดที่ต้องการประมวลผล
 - ที่ส่วนหัวของการประกาศฟังก์ชัน (Function header) ในส่วนของการประกาศพารามิเตอร์ หากพารามิเตอร์เป็นตัวแปรชุด 2 มิติ ให้ระบุเฉพาะจำนวนสมาชิกของมิติที่สอง (จำนวนคอลัมน์)
 - พารามิเตอร์ที่ใช้รับค่าของตัวแปรชุด อาจจะถูกประกาศในลักษณะของตัวแปรอยู่นั่นเองได้



ตัวอย่าง

- จะเขียนฟังก์ชัน `Is_IdentityMatrix()` ซึ่งทำการตรวจสอบเมทริกซ์ขนาด $n \times n$ ว่าเป็นเมทริกซ์เอกลักษณ์หรือไม่ หากใช่จะให้ผลเป็นจริง มิฉะนั้นแล้วจะให้ผลเป็นเท็จ
- วิเคราะห์



204112

53

#define N 10

#define True 1

#define False 0

```

int Is_IdentityMatrix(int x[][N], int n) {
    int i,j, ans=True;

    for (i=0; i<n && ans; i++)
        for (j=0; j<n && ans; j++)
            if (i==j)
                ans = ans && ((x[i][j]==1)?1:0);
            else
                ans = ans && ((x[i][j]==0)?1:0);
    return ans;
}
  
```

```

void DisplayMatrix(int x[][N], int n) {
    int i, j;
  
```

```

        for (i=0; i<n; i++) {
            for (j=0; j<n; j++)
                printf("%8d\t", x[i][j]);
            putchar('\n');
        }
    }
  
```

```

int main(void) {
    int x[N][N]={ {1,0,0}, {0,1,0}, {0,0,1} };
  
```

```

    DisplayMatrix(x, 3);
    if (Is_IdentityMatrix(x, 3))
        puts("is identity matrix...");
    else
        puts("is NOT identity matrix..");
    return 0;
}
  
```

204111



Passing Arrays to Function

```

/* Echo argument using pointer indexing */

#include <stdio.h>

main (int argc, char *argv[ ]) {
    /* argc is an argument count, argv is the list of argument */

    char **last_ptr = argv+argc-1;
    while (argv++<last_ptr)
        puts*argv);
}
  
```

204112

55

```

/* Echo argument using pointer indexing */

#include <stdio.h>

main (int argc, char *argv[ ]) {
  
```

int i;

```

    while (argv++<last_ptr)
        puts*argv);
}
  
```

204112

56