หน่วยเรียนที่ 6

ตัวแปรอาเรย์

ตัวแปรอาเรย์ (array) คือตัวแปรที่ประกาศเป็นชุดตัวแปร หรือตัวแปรที่มีลักษณะเป็นกลุ่มตัวแปร ซึ่งมีชื่อและชนิดข้อมูลของตัวแปรเดียวกัน โดยใช้ตัวเลขเรียงลำดับเป็นตัวกำกับเพื่ออ้างอิงชื่อสมาชิกของตัวแปรอาเรย์ การเรียกใช้ตัวแปรหรือสมาชิกใดในอาเรย์จะใช้ตัวเลขกำกับลำดับของอาเรย์เพื่อเข้าถึงข้อมูลนั้นๆ การจองหน่วยความจำจะใช้หน่วยความจำที่ใกล้เคียงกันในการจองหน่วยความจำให้กับสมาชิกในอาเรย์ลำดับถัดๆไป ซึ่งตัวแปรแบบอาเรย์มี 3 ลักษณะคือ อาเรย์แบบ 1 มิติ, อาเรย์แบบ 2 มิติ และอาเรย์แบบ 3 มิติ

6.1 ตัวแปรอาเรย์แบบ 1 มิติ

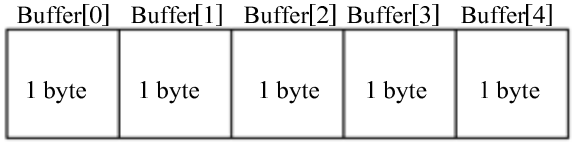
ตัวแปรอาเรย์แบบ 1 มิติ หมายถึงอาเรย์ ที่มีลักษณะของตัวแปรชุดที่เก็บข้อมูลแถวเดียว และอ้างถึงสมาชิกในกลุ่มโดยใช้ตัวเลขอ้างอิง ซึ่งมีลักษณะการประกาศตัวแปรแบบอาเรย์ดังนี้

ชนิดข้อมูล ชื่ออาเรย์ [n];

n คือ จำนวนสมาชิกตัวแปรอาเรย์ เช่น

char buffer[5];

หมายถึง ประกาศตัวแปรอาเรย์ชื่อ buffer เป็นข้อมูลชนิด char มีจำนวนสมาชิก 5 ตัว โดยมีชื่อสมาชิกเรียงลำดับตั้งแต่ buffer[0] ถึง buffer [4] ตัวแปรสมาชิกแต่ละตัวจะสามารถเก็บข้อมูลขนาด 8 บิตหรือ 1 ไบต์ตามขนาดชนิดข้อมูล char ที่ระบุไว้ดังแสดงในภาพที่ 6.1

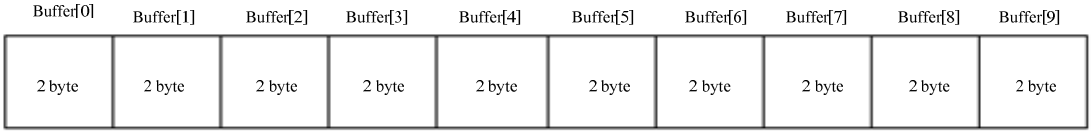


ภาพที่ 6.1 ลักษณะการจองหน่วยความจำเมื่อมีการประกาศตัวแปรขนาด 8 บิต ชนิดอาเรย์ 1 มิติ

หากต้องการให้สามารถเก็บข้อมูลได้ 16 บิต ต้องเปลี่ยนชนิดข้อมูลเป็น int เช่น

int buffer [10];

หมายถึง ประกาศตัวแปรอาเรย์ชื่อ buffer เป็นข้อมูลชนิด int มีจำนวนสมาชิก 10 ตัว โดยมีชื่อสมาชิกเรียงลำดับตั้งแต่ buffer[0] ถึง buffer [9] ตัวแปรสมาชิกแต่ละตัวจะสามารถเก็บข้อมูลขนาด 16 บิตหรือ 2 ไบต์ตามขนาดชนิดข้อมูล int ที่ระบุไว้ดังแสดงในภาพที่ 6.2



ภาพที่ 6.2 ลักษณะการจองหน่วยความจำเมื่อมีการประกาศตัวแปรขนาด 16 บิต ชนิดอาเรย์ 1 มิติ

การเรียกใช้ตัวแปรอาเรย์ คือการเรียกใช้สมาชิกในตัวแปรอาร์ย์ หลังจากได้ประกาศตัวแปรแล้วโดยใช้ตัวเลขกำกับชื่อสมาชิกในอาเรย์ ซึ่งสามารถเขียนคำสั่งได้ดังนี้

```
buffer[0] = 125;          // กำหนดให้ buffer[0] เก็บตัวเลขค่า 125
buffer[1] = 130;          // กำหนดให้ buffer[1] เก็บตัวเลขค่า 130
buffer[4] = 140;          // กำหนดให้ buffer[4] เก็บตัวเลขค่า 140
```

การกำหนดค่าให้กับอาเรย์ สามารถกำหนดค่าข้อมูลขณะประกาศอาเรย์ได้ด้วยเช่น

```
char num[10]  = {0,1,2,3,4,5,6,7,8,9};      // num[0] มีค่าเท่ากับ 0
                                             // num[3] มีค่าเท่ากับ 3
                                             // num [9] มีค่าเท่ากับ 9
```

ตัวอย่างที่ 6.1 โปรแกรมรับค่าจำนวนเต็ม 5 จำนวนจากผู้ใช้ และแสดงผลในลำดับที่กลับกัน

```
#include <stdio.h>
#define size 5
void main()
{
  int k;
int data[size];
```

```
clrscr();
for(k=0;k<size;k++)
{
  printf("\n Enter number %d :",k+1);
   scanf("%d",&data[k] );
}
printf("\nData Normal:");
for(k=0;k<size;k++)
{
  printf("\%d  ",data[k]);
  }
printf("\nData Swap :");
for(k=size-1;k>=0;k--)
{
   printf("\%d  ",data[k]);
 }
  getch();
}
```

เมื่อรันโปรแกรมแสดงข้อความให้ป้อนค่าตัวเลข จำนวน 5 ค่าข้อมูลด้วยกันแล้วจะนำไปเก็บไว้ในตัวแปรอาเรย์ data[0] ถึง data[4] เมื่อป้อนค่าครบจำนวนแล้ว จะมีการวนซ้ำนำค่าข้อมูลของสมาชิกอาเรย์ data[0] ถึง data[4] แสดงผลทางจอภาพตามลำดับ และมีการวนซ้ำนำข้อมูลของสมาชิกอาเรย์ data[4] ถึง data[0] แสดงผลทางจอภาพตามลำดับ ในลักษณะกลับลำดับข้อมูลดังในภาพที่ 6.3

```
Enter number 1: 10
Enter number 2:  20
Enter number 3:  30
Enter number 4:  40
Enter number 5:  50
Data Normal  :  10  20  30  40  50
Data Swap    :  50  40  30  20  10
```

ภาพที่ 6.3 ผลลัพธ์บนจอภาพของโปรแกรมตัวอย่างที่ 6.1

6.2 ตัวแปรอาเรย์แบบ 2 มิติ

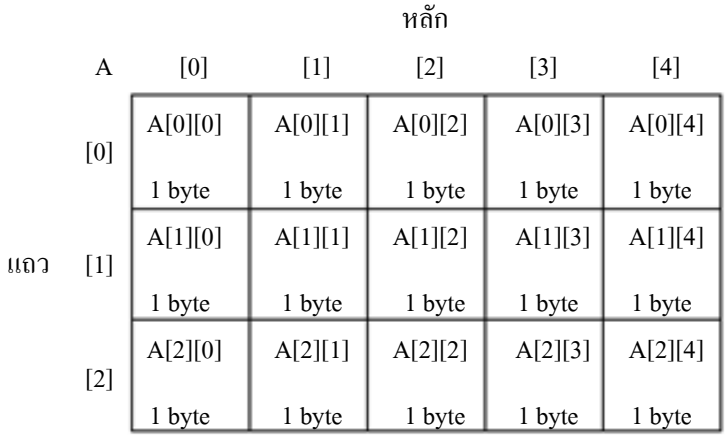
ตัวแปรอาเรย์แบบ 2 มิติ คือตัวแปรอาเรย์ที่มีลักษณะเป็นหลายแถวและหลายหลัก คล้ายกับการสร้างตาราง ที่มีทั้งด้านหลัก และด้านแถว ซึ่งทำให้ตัวแปรเดียวกันมีตำแหน่งที่สามารถเก็บข้อมูลของตัวแปรได้มากกว่า โดยลักษณะของการประกาศตัวแปรอาเรย์จะต้องระบุจำนวนแถวและหลักของอาเรย์ดังนี้

ชนิดข้อมูล ชื่ออาเรย์ [จำนวนแถว] [จำนวนหลัก];

เมื่อประกาศตัวแปรแบบอาเรย์แล้ว การเข้าถึงข้อมูลของตัวแปรอาเรย์ จะใช้ลำดับของแถว และหลัก เป็นตัวอ้างอิงตำแหน่ง เพื่อนำข้อมูลเข้าไปเก็บในตัวแปร หรือนำข้อมูลจากตัวแปรไปใช้งาน เช่น

```
char  A[3] [5];
```

หมายถึงประกาศตัวแปร A เป็นอาเรย์ 2 มิติ ขนาด 3 แถว และ 5 หลัก สมาชิกแต่ละตัวสามารถเก็บข้อมูลได้ 8 บิต ดังแสดงในภาพที่ 6.4



ภาพที่ 6.4 ลักษณะการจองหน่วยความจำเมื่อมีการประกาศตัวแปรขนาด 8 บิต ชนิดอาเรย์ 2 มิติ

การเรียกใช้สมาชิกของตัวแปรอาเรย์ แต่ละตัวโดยใช้ตัวเลขกำกับชื่อสมาชิกในอาเรย์ สามารถเขียนคำสั่งได้ดังนี้

```
A[0] [0]   = 10;          // กำหนดให้ A[0][0] เก็บตัวเลขค่า 10
A[0][1]    = 13;          // กำหนดให้ A[0][1] เก็บตัวเลขค่า 13
A[2] [4]   = 14;          // กำหนดให้ A[2][4] เก็บตัวเลขค่า 14
```

การกำหนดค่าให้กับอาเรย์ สามารถกำหนดค่าข้อมูลขณะประกาศอาเรย์ได้ด้วยเช่น

```
char num[3][3]  = {1,2,3,          // num[0][0] มีค่าเท่ากับ 1
                   4,5,6,          // num[2][3] มีค่าเท่ากับ 6
                   7,8,9};         // num[3][3] มีค่าเท่ากับ 9
```

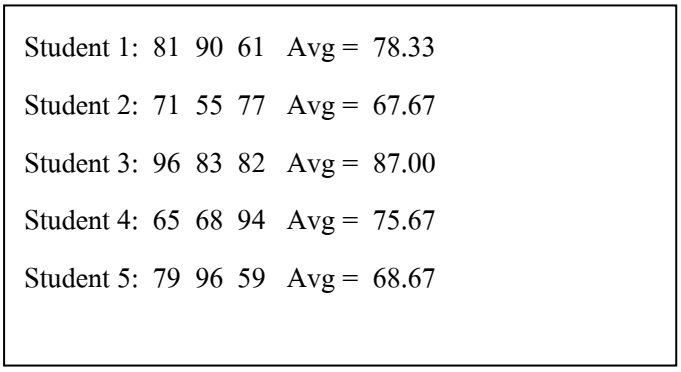
ตัวอย่างที่ 6.2 โปรแกรมหาค่าเฉลี่ยผลการสอบจำนวน 3 ครั้ง ของนักศึกษา จำนวน 5 คนโดยให้ด้านแถวเป็นจำนวนนักศึกษา และด้านหลักเป็นคะแนนสอบแต่ละครั้งของนักศึกษาดังตารางที่ 6.1

ตารางที่ 6.1 ผลคะแนนสอบของนักศึกษา จำนวน 5 คน

ผลการสอบ	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3
นักศึกษาคนที่ 1	84	90	61
นักศึกษาคนที่ 2	71	55	77
นักศึกษาคนที่ 3	96	83	82
นักศึกษาคนที่ 4	65	68	94
นักศึกษาคนที่ 5	79	96	59

```
#include <stdio.h>
void main()
{
  int i,j,sum,score[5][3] = {{84, 90, 61},    // ประกาศตัวแปรอาเรย์แบบ 2 มิติ
                             {71, 55, 77},    // พร้อมทั้งกำหนดค่าคะแนนสอบ
                             {96, 83, 82},
                             {65, 68, 94},
                             {79, 68, 59}};
clrscr();
for(i=0;i<5;i++)
{
  printf("\n Student %d:   ",i+1);
  for(j=0,sum=0;j<3;j++)
    {
          printf("%d  ",score[i][j]);
          sum = sum+score[i][j];
    }
    printf("Avg = %.2f ", (float)sum/3);
}
getch();
}
```

จากโปรแกรมมีการประกาศตัวแปรอาเรย์แบบ 2 มิติพร้อมทั้งกำหนดคะแนนผลสอบ 3 ครั้งของนักศึกษาจำนวน 5 คน จากนั้นจะวนซ้ำแสดงผลคะแนนสอบและคะแนนเฉลี่ยของแต่ละคน จนกครบ 5 คนดังแสดงในภาพที่ 6.5



ภาพที่ 6.5 ผลลัพธ์บนจอภาพของโปรแกรมตัวอย่างที่ 6.2

6.3 ตัวแปรอาเรย์แบบ 3 มิติ

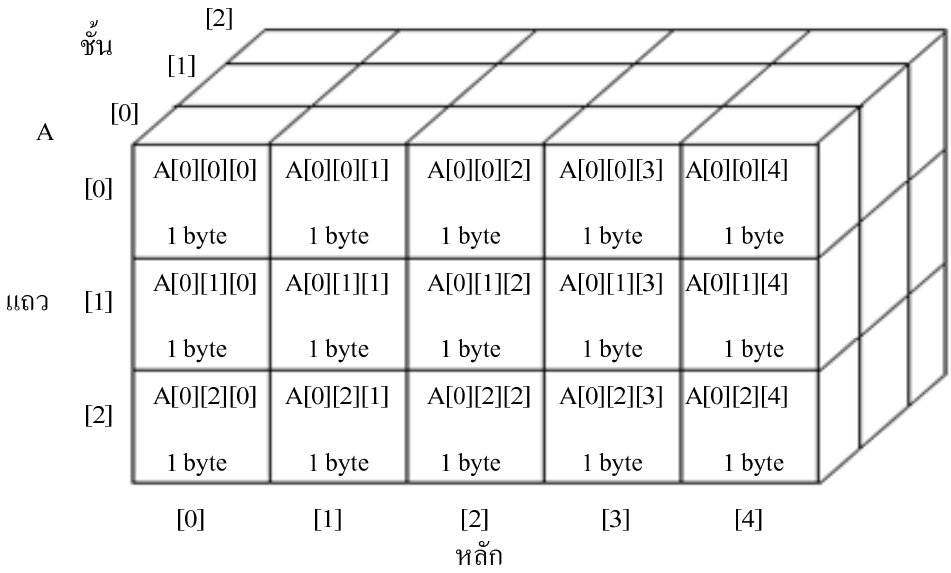
ตัวแปรอาเรย์แบบ 3 มิติ คือตัวแปรอาเรย์ที่มีลักษณะเป็นหลายแถวและหลายหลักและหลายชั้น ซึ่งคล้ายกับอาเรย์ 2 มิติ แต่เพิ่มลักษณะของชั้นหรือแผ่นซ้อนกันไป ซึ่งเปรียบเทียบได้กับสมุดตารางซึ่งมีอยู่หลายๆแผ่น โดยแต่ละแผ่นจะมีลำดับแถวและหลักกำกับอยู่คล้ายกับ 2 มิติ และมิติที่ 3 คือ ชั้นหรือแผ่นซ้อนอยู่ ซึ่งลักษณะของการประกาศตัวแปรอาเรย์จะต้องระบุจำนวนชั้น จำนวนแถวและจำนวนหลักของอาเรย์ 3 มิติดังนี้

ชนิดข้อมูล ชื่ออาเรย์ [จำนวนชั้น] [จำนวนแถว] [จำนวนหลัก];

เช่น

```
char  A[3][4] [5];
```

หมายถึงประกาศตัวแปร A เป็นอาเรย์ขนาด 3 มิติ โดยมีขนาด 3 ชั้น 4 แถว และ 5 หลัก สมาชิกแต่ละตัวสามารถเก็บข้อมูลได้ 8 บิต ดังแสดงในภาพที่ 3.6



ภาพที่ 6.6 ลักษณะการจองหน่วยความจำเมื่อมีการประกาศตัวแปรขนาด 8 บิต ชนิดอาเรย์ 3 มิติ

การเรียกใช้ตัวแปรอาเรย์ ของสมาชิกแต่ละตัวโดยใช้ตัวเลขกำกับชื่อสมาชิกในอาเรย์ สามารถเขียนคำสั่งได้ดังนี้

```
A[0][0] [0]   = 32;          // กำหนดให้ A[0][0][0] เก็บตัวเลขค่า 32
A[1][0][1]    = 45;          // กำหนดให้ A[1][0][1] เก็บตัวเลขค่า 45
A[2][3] [4]   = 65;          // กำหนดให้ A[2][3][4] เก็บตัวเลขค่า 65
A[1][1][1]    = A[2][3][4]   // กำหนดให้ A[1][1][1] เก็บข้อมูลที่อยู่ใน
A[2][3][4]  ซึ่งก็คือ        // ตัวเลข 6
```

6.4 ตัวแปรอาเรย์อักขระ

ตัวแปรอาเรย์อักขระ หรือ ตัวแปรสตริง (string) ที่นำตัวแปรอาเรย์ ชนิดข้อมูลอักขระแบบอาเรย์ 1 มิติมาประยุกต์ใช้เป็นตัวแปรแบบสตริง เนื่องจากภาษาซีจะไม่มีชนิดข้อมูลแบบสตริง โดยตรงเหมือนกับภาษาปาสคาล จะมีเพียงชนิดข้อมูลอักขระ (Character) เท่านั้น ดังนั้นในการสร้างตัวแปรแบบสตริงจะเป็นการรวมอักขระ (char) ให้อยู่ในรูปแบบของอาเรย์ของอักขระ เช่น ประกาศตัวแปร

```
char  name[20];          // ตั้งชื่อตัวแปร name เป็นสตริงขนาด 20 อักขระ
```

หมายถึงเป็นอาเรย์ขนาด 20 สมาชิก สามารถเก็บอักขระได้ 20 ตัว และสมาชิกตัวสุดท้ายจะเก็บสัญลักษณ์ \0 (null character) โดยเครื่องหมาย \0 ที่ปิดท้ายโปรแกรมจะถูกกำหนดให้โดยอัตโนมัติ หรือผู้เขียนโปรแกรมสามารถกำหนดได้เช่น

```
char name[15];              // ตั้งชื่อตัวแปร name เป็นสตริงขนาด 15  อักขระ
char  msg[5][10];           // ตัวแปรสตริง msg จำนวน 5 ตัวแต่ละตัวมีขนาด 10 อักขระ
char  msg1[5]  = “abcd”;  // ตัวแปร msg1 เก็บอักขระ ‘a’,’b’,’c’,’d’,’\0’
char msg2[8] ={'C','O','M','P','U','T','E','R'};
char msg3[] = "Hello, World";
```

ลักษณะของตัวแปรอาเรย์ msg3[] จะมีการจองพื้นที่ขนาด 13 ตัวอักขระรวมทั้ง \0 โดยการเก็บค่าจะเหมือนกับตัวแปรอาเรย์ 1 มิติ เก็บค่าอักขระ โดยตัวแปรอาเรย์ msg3[0] = 'H' ,msg3[1] = 'e' , msg3[2] = 'l' และอาเรย์ถัดไปก็เก็บอักขระถัดไป ดังในภาพที่ 6.7

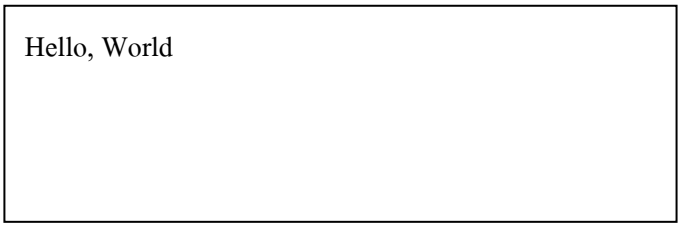
msg3	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]
	H	e	l	l	o	,		W	o	r	l	d	\0

ภาพที่ 6.7 ตำแหน่งอาเรย์ที่ใช้เก็บข้อความอักขระ

ตัวอย่างที่ 6.3 โปรแกรมแสดงข้อความจากตัวแปรอาเรย์อักขระ

```
#include <stdio.h>
void main()
{
  char msg3[] = "Hello, World";
  int i;
  for(i=0;i<13;i++)
  printf("%C",msg3[i]);
}
```

จากโปรแกรมจะมีการประกาศตัวแปรอาเรย์ msg3[] พร้อมทั้งกำหนดข้อมูลอักขระ "Hello, World" ซึ่งจะทำให้ตัวแปรอาเรย์ msg3[] มีสมาชิกจำนวน 12 สมาชิก จากนั้นจะมีการวนซ้ำ แสดงอักขระที่เป็นสมาชิกของตัวแปรอาเรย์ ทีละตัวจนเป็นข้อความดังในภาพที่ 6.8



ภาพที่ 6.8 ผลลัพธ์บนจอภาพของโปรแกรมตัวอย่างที่ 6.3

นอกจากนี้ซีคอมไพเลอร์ได้จัดเตรียมฟังก์ชันมาตรฐานสำหรับจัดการข้อมูลที่เกี่ยวข้องกับตัวแปรที่มีลักษณะเป็นสตริง เช่นการนับจำนวนตัวอักษรในตัวแปรอาเรย์ การค้นหา การเปรียบเทียบเป็นต้น โดยเป็นฟังก์ชันมาตรฐานที่อยู่ในไลบรารี string.h ซึ่งจะต้องประกาศ #include<string.h> ที่ส่วนหัวโปรแกรมเมื่อต้องการใช้งานฟังก์ชันที่อยู่ภายในไฟล์ string.h ฟังก์ชันที่เกี่ยวข้องกับข้อความสตริงซึ่งใช้งานทั่วๆไปมีดังตารางที่ 6.2

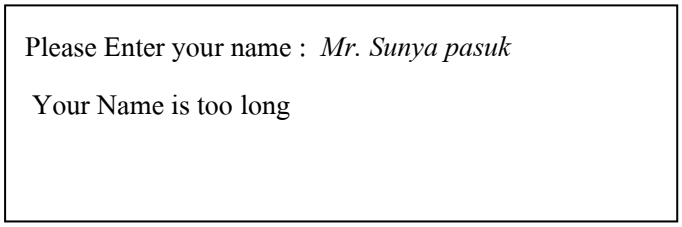
ตารางที่ 6.2 ฟังก์ชันมาตรฐานกับตัวแปรสตริง

ฟังก์ชันและตัวอย่าง	**ความหมายและการใช้งาน**
strlen() len = strlen(s1);	หาค่าจำนวนของตัวอักษรสตริง s1
strcat() ptr = strcat(s1,s2);	เพิ่มข้อความสตริง s2 เข้ากับข้อความสตริง s1
strchr() ptr = strchar(s1,c);	หาตัวอักขระ c ในสตริง s1 เพื่อพบส่งค่าตำแหน่ง &s1[i] กลับ
strrchr() ptr = strrchar(s1,c);	ทำงานแบบเดียวกับฟังก์ชัน strchr() แต่ค้นหาในทิศทางตรงข้าม
strcmp() result = strcmp(s1,s2);	เปรียบเทียบข้อความสตริง s1 กับสตริง s2 ส่งค่ากลับดังนี้ ถ้า s1< s2 ส่งค่าน้อยกว่า 0 กลับมา ถ้า s1 == s2 ส่งค่า 0 กลับมา ถ้า s1 > s2 ส่งค่ามากกว่า 0 กลับมา
strncmp() result = strncmp(s1,s2,n);	เปรียบเทียบสตริง s1 กับสตริง s2 จำนวน n ตัวอักขระ และส่งค่ากลับเหมือนฟังก์ชัน strcmp()
stricmp() result = stricmp(s1,s2);	เปรียบเทียบสตริง s1 กับสตริง s2 โดยไม่สนใจอักษรเล็กหรือตัวอักษรใหญ่

ตัวอย่างที่ 6.4 ตรวจสอบความยาวของข้อความ

```
#include <stdio.h>
#include <string.h>
void main()
{
  char  read_in_string[10],              // จองหน่วยความจำสำหรับตัวแปรอาเรย์
        copy_of_string[10];
  printf("Please Enter Your Name : ");
  gets(read_in_string);                  // รับข้อความสตริงจากคีย์บอร์ด
  strcpy(copy_of_string,read_in_string);  // คัดลอกข้อความจากอาเรย์
                                         // read_in_string ไปยัง copy_of_string
  if(strlen(copy_of_string)>10)
    printf("\n Your Name is too long");
 else
    printf("%s",copy_of_string);        // พิมพ์ข้อความจากอาเรย์ copy_of_string
}
```

ผลการทำงานของโปรแกรมจะให้เราป้อนชื่อ และนำชื่อข้อความที่ป้อนเก็บไว้ในตัวแปรอาเรย์และตรวจสอบความยาวของชื่อที่ป้อนด้วยฟังก์ชัน strlen(copy_of_string) หากชื่อที่ป้อนมีตัวอักขระมากกว่า 10 ตัว if(strlen(copy_of_string)>10) ก็จะแสดงข้อความ Your Name is too long หากชื่อที่ป้อนอักขระไม่เกิน 10 ตัวจะแสดงชื่อที่ป้อนบนจอภาพดังในภาพที่ 6.9



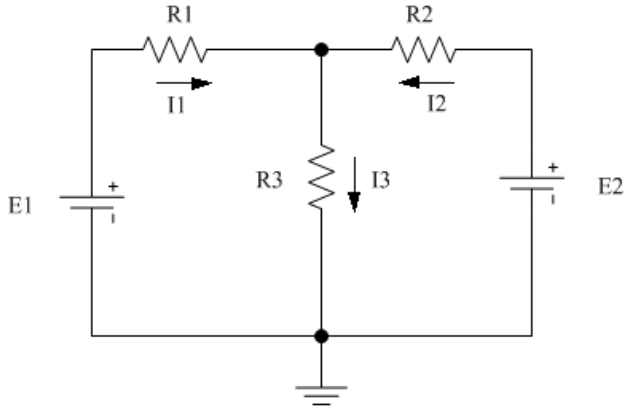
ภาพที่ 6.9 ผลลัพธ์บนจอภาพของโปรแกรมตัวอย่างที่ 6.4

แบบฝึกหัดท้ายหน่วยเรียนที่ 6

1) จงเขียนโปรแกรมเพื่อรับข้อมูล N จำนวน โดยเก็บข้อมูลในตัวแปรอาเรย์ และให้พิมพ์ข้อมูลในลำดับที่กลับกับการป้อนข้อมูล เช่นป้อนข้อมูล 1 2 3 4 5 แต่จะแสดงคำตอบในลักษณะเรียงกลับเป็น 5 4 3 2 1

2) จงเขียนโปรแกรมเพื่อรับค่าข้อมูลเมตริกขนาด 2x2 และหาค่าดีเทอร์มิแนนท์ของเมตริกและแสดงการเรียงค่าสมาชิกเป็นแบบเมตริก

3) จงเขียนโปรแกรมหาคำตอบของวงจรไฟฟ้า หาค่า I1, I2 และ I3 ในภาพที่ 6.10 โดยใช้ทฤษฎีการแก้ปัญหาของเคอร์ชอฟ โดยรับค่า R1, R2, R3, E1 และ E2



ภาพที่ 6.10 วงจรไฟฟ้าเพื่อแก้ปัญหาด้วยทฤษฎีเคอร์ชอฟ

4) จงเขียนโปรแกรมเพื่อรับค่าข้อมูลการวัดอุณหภูมิของร่างกายคนไข้เป็นเวลา 7 วัน โดยแต่ละวันจะทำการวัดอุณหภูมิ 4 ครั้งที่เวลา 6.00 น. 12.00น. 18.00 น. และ 24.00น. โดยจะต้องเขียนโปรแกรมเพื่อรับค่าข้อมูลแต่ละวันและแต่ละครั้งและโปรแกรมสามารถคำนวณหาค่าอุณหภูมิสูงสุดและ ต่ำสุดในแต่ละวัน หาค่าอุณหภูมิเฉลี่ยที่วัดได้ในแต่ละวัน หาค่าอุณหูมิเฉลี่ยในแต่ละเวลาทั้ง 7 วัน และอุณหภูมิเฉลี่ยทั้งหมด

ใบงานที่ 10

ตัวแปรอาเรย์ 2 มิติ

จุดประสงค์

1) ทดลองเขียนโปรแกรมโดยใช้ตัวแปรอาเรย์ 2 มิติ
2) สังเกตเขียนโปรแกรมใช้ตัวแปรอาเรย์แก้ปัญหาทางคณิตศาสตร์

ตัวแปรอาเรย์ (array) คือตัวแปรที่ประกาศเป็นชุดตัวแปร หรือตัวแปรที่มีลักษณะเป็นกลุ่มตัวแปร ซึ่งมีชื่อและชนิดข้อมูลของตัวแปรเดียวกันโดยใช้ตัวเลขเรียงลำดับเป็นตัวกำกับเพื่ออ้างอิงชื่อสมาชิกของตัวแปรอาเรย์ การเรียกใช้ตัวแปรหรือสมาชิกใดในอาเรย์จะใช้ตัวเลขกำกับลำดับของอาเรย์เพื่อเข้าถึงข้อมูลนั้นๆ การจองหน่วยความจำจะใช้หน่วยความจำที่ใกล้เคียงกันในการจองหน่วยความจำให้กับสมาชิกในอาเรย์ลำดับถัดๆไป ซึ่งตัวแปรแบบอาเรย์มี 3 ลักษณะคือ

1) อาเรย์แบบ 1 มิติ
2) อาเรย์แบบ 2 มิติ
3) อาเรย์แบบ 3 มิติ

ตัวแปรอาเรย์แบบ 2 มิติ คือตัวแปรอาเรย์ที่มีลักษณะเป็นหลายแถวและหลายหลักคล้ายกับการสร้างตาราง ที่มีทั้งด้านหลัก และด้านแถว ซึ่งทำให้ตัวแปรเดียวกันมีตำแหน่งที่สามารถเก็บข้อมูลของตัวแปรได้มากกว่า โดยลักษณะของการประกาศตัวแปรอาเรย์จะต้องระบุจำนวนแถวและหลักของอาเรย์ดังนี้

ชนิดข้อมูล ชื่ออาเรย์ [จำนวนแถว] [จำนวนหลัก];

การทดลองที่ 10.1 การใช้ตัวแปรอาเรย์ 2 มิติ

การใช้ตัวแปรอาเรย์ 2 มิติจะมีความคล้ายกับการสร้างตารางที่มีจำนวนแถวและจำนวนหลักสำหรับเก็บค่าข้อมูล โดยใช้ลำดับตัวเลขหลักและแถวเป็นตัวกำหนดตำแหน่งที่ใช้สำหรับเก็บข้อมูล ซึ่งในการทดลองนี้ได้สร้างตารางสำหรับเก็บผลการสอบของนักศึกษาทั้ง 3 ครั้ง แล้วหาค่าเฉลี่ยของการทดสอบ 3 ครั้ง ของจำนวนนักศึกษาที่สามารถกำหนดได้แต่ทั้งนี้ไม่เกินจำนวน 50 คน โดยมีขั้นตอนการทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่

2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include <stdio.h>
void main()
{
  int i,j,sum,num,score[50][3];
 clrscr();
 printf("\n ************************************");
 printf("\n *          Find Average score 3 Test          *");
 printf("\n ************************************");
 printf("\n Enter number of student : ");
 scanf("%d",&num);
 printf("\n Enter the score of student : ");
for(i=0;i<num;i++)
{
  printf("\n Student %d:   ",i+1);
  for(j=0,sum=0;j<3;j++)
    {
            printf("\n Enter score of Student %d Time %d : ",i+1,j+1);
            scanf("%d",&score[i][j]);
              }
    }
```

```
for(i=0;i<num;i++)
  {
    printf("\n Student %d:   ",i+1);
     for(j=0,sum=0;j<3;j++)
     {
                printf("%d  ",score[i][j]);
                sum = sum+score[i][j];
     }
     printf("Avg = %.2f ", (float)sum/3);
}
getch();
}
```

3) บันทึกไฟล์ในเป็นไฟล์ program10_1.c

4) เลือกคำสั่ง Compile และให้บันทึกผล

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

……

……

……

……

……

การทดลองที่ 10.2 การหาค่าดิเทอมิแนนท์ของเมตริกซ์

การเขียนโปรแกรมคอมพิวเตอร์มักจะเกี่ยวข้องกับการคำนวณทางคณิตศาสตร์ ซึ่งสามารถเขียนโปรแกรมเพื่อหาคำตอบทางคณิตศาสตร์ได้ โดยในการทดลองนี้จะเป็นการเขียนโปรแกรมเพื่อหาค่าดีเทอร์มิแนนท์ของเมตริก A ขนาด 3 x 3 โดยให้ผู้ใช้ป้อนค่าข้อมูลภายในของเมตริกซ์

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

$$\det(A) = (A_{11}xA_{22}xA_{33}) + (A_{12}xA_{23}xA_{31}) + (A_{13}xA_{21}xA_{32}) - (A_{31}xA_{22}xA_{13}) - (A_{32}xA_{23}xA_{11}) - (A_{33}xA_{21}xA_{12})$$

ซึ่งสามารถใช้ตัวแปรอาเรย์เก็บข้อมูลของสมาชิกภายในอาเรย์และมีขั้นตอนการทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่

2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include<stdio.h>
#include<conio.h>
int A[3][3];
void get_value_matrix(void)
{
   int i,j;
  for(i=1;i<=3;i++)
  {
    for(j=1;j<=3;j++)
    {
              printf("\n Enter value Matrix A%d,%d = ",i,j);
              scanf("%d",&A[i][j]);
```

```
      }
    }
  }
float det_A(void)
{
float deter;
deter = ((A[1][1]* A[2][2] *A[3][3] + A[1][2]*A[2][3]*A[3][1] +
          A[1][3]*A[2][1]*A[3][2]) - (A[3][1]*A[2][2]*A[1][3] +
          A[3][2]*A[2][3]*A[1][1]+A[3][3]*A[2][1]*A[1][2]));
return(deter);
}

void show_matrix(void)
{
 int i,j;
   for(i=1;i<=3;i++)
  {
     printf("\n      | ");
      for(j=1;j<=3;j++)
     {
        printf("%d   ",A[i][j]);
     }
    printf("|");
   }
}
void main()
{
  float A;
   clrscr();
   printf(" *********************************\n");
```

```
  printf("*       Find Determinant  Matrix A          *\n");
  printf("*********************************\n");
  printf("\n");
  printf("\n");
  get_value_matrix();
  show_matrix();
  A = det_A();
 printf("\n\n Determinant A =%.2f ",A);
getch();
}
```

3) บันทึกไฟล์ในเป็นไฟล์ program10_2.c

4) เลือกคำสั่ง Compile

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

……

……

……

……

……

6) สรุปผลการทดลอง

……
……
……
……
……
……
……
……
……
……

งานที่มอบหมาย

1) จงเขียนโปรแกรมตามโจทย์ข้อ 3 ในแบบฝึกหัดที่ท้ายบทเรียนที่ 6 โดยมีการสร้างเป็นฟังก์ชันไว้ด้วย

ใบงานที่ 11

การประยุกต์ตัวแปรอาเรย์กับการคำนวณเมตริกซ์

จุดประสงค์

1) ทดลองเขียนโปรแกรมประยุกต์กับการคำนวณเมตริกซ์
2) สังเกตการเขียนโปรแกรมประยุกต์ใช้ตัวแปรอาเรย์แก้ปัญหาโจทย์ทางคณิตศาสตร์

เมตริกซ์ คือกลุ่มของจำนวนหรือสมาชิกของจำนวนจริงใดๆ ที่เขียนเรียงกันเป็นเป็นแถวและเรียงกันในแนวตั้ง เช่น

$$\begin{bmatrix} 1 & 5 & 3 \\ 0 & 1 & 4 \\ 5 & -3 & -4 \end{bmatrix}$$

ภาพที่ 6.11 สมาชิกและรูปแบบของเมติกซ์ขนาด 3x3

เราเรียกแถวในแนวนอนของเมตริกซ์ว่า แถว เรียกแถวในแนวตั้งของเมตริกซ์ว่า หลัก และเรียกจำนวนแต่ละจำนวนในเมตริกซ์ว่า สมาชิก ของเมตริกซ์ การกล่าวถึงสมาชิกของเมตริกซ์จะต้องระบุตำแหน่งแถว และหลัก เช่นสมาชิกที่อยู่ในแถวที่ 2 หลักที่ 3 คือเลข 4 สมาชิกที่อยู่แถวที่ 2 หลักที่ 2 คือเลข 1 เป็นต้น ขนาดของเมตริกซ์จะขึ้นอยู่กับจำนวนแถวและจำนวนหลักของเมตริกซ์ เช่นเมตริกซขนาด m แถว และ n หลัก เรียกเมตริกซ์ขนาด m x n เช่นการกล่าวถึงเมตริกซ์ A ขนาด m x n จะใช้สัญลักษณ์ $A_{m\,x\,n}$ และสมาชิกภายในเมตริกซ์ จะใช้สัญลักษณ์ $a_{i,j}$ แทนตำแหน่งของสมาชิก โดย i หมายถึงลำดับแถว และ j หมายถึงลำดับหลักของเมตริกซ์ดังภาพที่ 6.12

$$A = A_{m\times n} = \begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & \cdots & a_{2n} \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & \cdots & a_{mn} \end{bmatrix}$$

ภาพที่ 6.12 ลำดับแถวและหลักของสมาชิกเมตริกซ์

การทดลองที่ 11.1 การบวกเมตริกซ์

การบวกเมตริกซ์โดยทั่วไปจะนิยามให้เมทริกซ์สองเมตริกซ์มีมิติเท่ากัน ผลบวกของเมตริกซ์ A และ B ที่มีมิติ $m \times n$ เขียนแทนด้วย $A + B$ และได้ผลลัพธ์ออกมาเป็นเมตริกซ์ขนาด $m \times n$ ที่มีสมาชิกเป็นผลบวกบนตำแหน่งที่ตรงกัน ตัวอย่างเช่น

$$\begin{bmatrix} 1 & 3 \\ 1 & 0 \\ 1 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 7 & 5 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 1+0 & 3+0 \\ 1+7 & 0+5 \\ 1+2 & 2+1 \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ 8 & 5 \\ 3 & 3 \end{bmatrix}$$

ภาพที่ 6.13 ผลการบวกเมตริกซ์ขนาด 3x2

โดยหากต้องการให้โปรแกรมรับค่าสมาชิกแต่ละตัวของเมตริกซ์ A และ B ขนาด 3x3 แล้วทำการบวกเมตริกซ์ A และ B แล้วเก็บในเมตริกซ์ดังสมการ C = A + B มีขั้นตอนทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่

2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include<stdio.h>
#include<conio.h>
int A[3][2];
int B[3][2];
int C[3][2];
void get_value_matrix_A(void)
{
   int i,j;
  for(i=1;i<=3;i++)
  {
    for(j=1;j<=2;j++)
    {
              printf("\n Enter value Matrix A%d,%d = ",i,j);
              scanf("%d",&A[i][j]);
     }
```

```
    }
  }
void get_value_matrix_B(void)
{
   int i,j;
  for(i=1;i<=3;i++)
  {
    for(j=1;j<=2;j++)
    {
                printf("\n Enter value Matrix B%d,%d = ",i,j);
                scanf("%d",&B[i][j]);
     }
   }
  }
void Add_matrix_AB(void)
{
   int i,j;
  for(i=1;i<=3;i++)
  {
    for(j=1;j<=2;j++)
    {
                C[i][j] = A[i][j] +B[i][j] ;
     }
   }
  }
void Result_matrix_C(void)
{
   int i,j;
   printf("\n Matrix C =  A + B ");
   printf("\n");
```

```
  for(i=1;i<=3;i++)
  {
            if(i==2)
             printf("\n | %d  %d  | + | %d  %d  | = | %d  %d  |"
                                    ,A[i][1],A[i][2],B[i][1],B[i][2],C[i][1],C[i][2]);
            else
            printf("\n | %d  %d  |     | %d  %d  |   | %d  %d  |"
                                    ,A[i][1],A[i][2],B[i][1],B[i][2],C[i][1],C[i][2]);
  }
 }
void main()
{
   clrscr();
  printf(" **********************************\n");
  printf("*      Find Add  Matrix A +B              *\n");
  printf("**********************************\n");
  printf("\n");
  printf("\n");
  get_value_matrix_A();
  get_value_matrix_B();
  Add_matrix_AB();
 Result_matrix_C
 getch();
}
```

3) บันทึกไฟล์ในเป็นไฟล์ program11_1.c

4) เลือกคำสั่ง Compile

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

การทดลองที่ 11.2 การคูณเมตริกซ์

การคูณเมตริกซ์ด้วยเมตริกซ์ ถ้าหากเมตริกซ์ A และ B เป็นเมตริกซ์ใดๆ การนำเมตริกซ์ A มาคูณกับเมตริกซ์ B ได้นั้นเมตริกซ์ต้องเป็นเมตริกซ์ขนาด $A_{(m \times p)}$ และเมตริกซ์ $B_{(q \times n)}$ ซึ่งขนาด p = q และจะทำให้ได้เมตริกซ์ขนาด m x n ถ้า $A = (a_{i,j})_{m \times p}$ และ $B = (b_{i,j})_{qxn}$ เป็นเมตริกซ์สองเมตริกซ์ โดยที่จำนวนหลักของเมตริกซ์ A เท่ากับจำนวนแถวของเมตริกซ์ B แล้ว เราสามารถนิยาม *ผลคูณ AB* ว่าเป็นเมตริกซ์ $C = (c_{i,j})_{mxn}$ โดยที่

$$c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,n}b_{n,j} = \sum_{k=1}^{n} a_{i,k}b_{k,j}$$

กล่าวคือสมาชิกในแถว i หลัก j ของผลคูณเมตริกซ์ AB คำนวณได้จากการนำสมาชิกของหลัก i ของเมตริกซ์ A และสมาชิกของคอลัมน์เมตริกซ์ B ในตำแหน่ง "เดียวกัน" มาคูณกัน แล้วนำผลคูณทั้งจำนวน n ผลคูณนั้นมาบวกกัน โดยการทดลองนี้จะให้เมตริกซ์ A เป็นเมตริกซ์ $A_{(3x2)}$ และเมตริกซ์ B เป็นเมตริกซ์ $B_{(2x3)}$ โดยจะเป็นการเขียนโปรแกรมรับค่าสมาชิกของเมตริกซ์ A และ B แล้วนำเมตริกซ์ A และ B มาคูณกัน ซึ่งมีลำดับขั้นตอนการทดลองดังนี้

1) เปิดโปรแกรมคอมไพเลอร์ Turbo C และ สร้างไฟล์ใหม่

2) เขียนโปรแกรมลงในอีดิตเตอร์ตามตัวอย่างโปรแกรมดังนี้

```
#include<stdio.h>
#include<conio.h>
int A[3][2];
int B[2][3];
int C[3][3];
void get_value_matrix_A(void)
{
   int i,j;
  for(i=1;i<=3;i++)
  {
    for(j=1;j<=2;j++)
    {
```

```
                printf("\n Enter value Matrix A%d,%d = ",i,j);
                scanf("%d",&A[i][j]);
      }
    }
  }
void get_value_matrix_B(void)
{
   int i,j;
  for(i=1;i<=2;i++)
  {
    for(j=1;j<=3;j++)
    {
                printf("\n Enter value Matrix B%d,%d = ",i,j);
                scanf("%d",&B[i][j]);
     }
   }
  }
void Multi_matrix_AB(void)
{
   int i,j;
  C[1][1] = A[1][1] *B[1][1] + A[1][2]*B[2][1] ;
  C[1][2] = A[1][1] *B[1][2] + A[1][2]*B[2][2] ;
  C[1][3] = A[1][1] *B[1][3] + A[1][2]*B[2][3] ;
  C[2][1] = A[2][1] *B[1][1] + A[2][2]*B[2][1] ;
  C[2][2] = A[2][1] *B[1][2] + A[2][2]*B[2][2] ;
  C[2][3] = A[2][1] *B[1][3] + A[2][2]*B[2][3] ;
  C[3][1] = A[3][1] *B[1][1] + A[3][2]*B[2][1] ;
  C[3][2] = A[3][1] *B[1][2] + A[3][2]*B[2][2] ;
  C[3][3] = A[3][1] *B[1][3] + A[3][2]*B[2][3] ;
  }
```

```
void Result_matrix_C(void)
{
   int i,j;
   printf("\n Matrix C =  A x B ");
   printf("\n");
   for(i=1;i<=3;i++)
   {
                if(i<3)
                 printf("\n |  %d   %d |   x   |   %d   %d    %d  |  =  | %d   %d  %d  |"
                  ,A[i][1],A[i][2],B[i][1],B[i][2],B[i][3],C[i][1],C[i][2],C[i][3]);
                else
                 printf("\n |  %d   %d |                          | %d   %d  %d  |"
                 ,A[i][1],A[i][2],C[i][1],C[i][2],C[i][3]);
   }
  }
 main()
{
   //clrscr();
  printf(" *********************************\n");
  printf("*       Find Multiple Matrix A x B              *\n");
  printf("*********************************\n");
  printf("\n");
  printf("\n");
  get_value_matrix_A();
  get_value_matrix_B();
  Multi_matrix_AB();
  Result_matrix_C();
  getch();
}
```

3) บันทึกไฟล์ในเป็นไฟล์ program11_2.c

4) เลือกคำสั่ง Compile

5) เลือกคำสั่ง RUN และให้บันทึกผลการทดลอง

……
……
……
……
……
……
……
……
……

6) สรุปผลการทดลอง

……
……
……
……
……
……
……

งานที่มอบหมาย

1) จงเขียนโปรแกรมรับค่าเมตริกซ์ขนาด 3 x 3 และคำนวณหาค่าทรานโพสของเมตริกซ์