บทที่ 6 อาร์เรย์ (Array)

ตัวแปรชุด (Arrays) คือ กลุ่มของข้อมูลที่มีชนิดของข้อมูลเหมือนกัน จึงทำการจัดกลุ่มไว้ด้วยกัน แล้วอ้างถึงกลุ่มข้อมูลนั้นนั้นด้วยชื่อเดียว และอ้างถึงสมาชิกแต่ละตัวในกลุ่มของตัวแปรชุดนั้นด้วยหมายเลข

ตัวอย่างของปัญหาที่จำเป็นต้องใช้งานตัวแปรชุดเช่น หากต้องการรับข้อมูลคะแนนสอบของนักศึกษาแต่ละคนที่ลงเรียนในวิชาหนึ่งซึ่งมีนักศึกษาลงเรียน 40 คน ต้องการทราบว่ามีนักศึกษาที่ได้คะแนนต่ำกว่าคะแนนเฉลี่ยกี่คน ในกรณีเช่นนี้จะเห็นว่ามีการรับข้อมูลคะแนนสอบของนักศึกษาแต่ละคน จากนั้นนำมาหาคะแนนเฉลี่ย และจะต้องมีการนำข้อมูลคะแนนสอบของนักศึกษาแต่ละคนมาเปรียบเทียบกับคะแนนเฉลี่ยว่าจะมีนักศึกษาได้คะแนนน้อยกว่าคะแนนเฉลี่ยกี่คน การทำงานในลักษณะที่ต้องมีการเก็บข้อมูลที่เหมือน ๆ กัน และนำกลับมาใช้เพื่อประมวลผลงานใดงานหนึ่งเช่นนี้ จำเป็นต้องใช้ตัวแปรชุดมาช่วยในการเก็บข้อมูลเสมอ

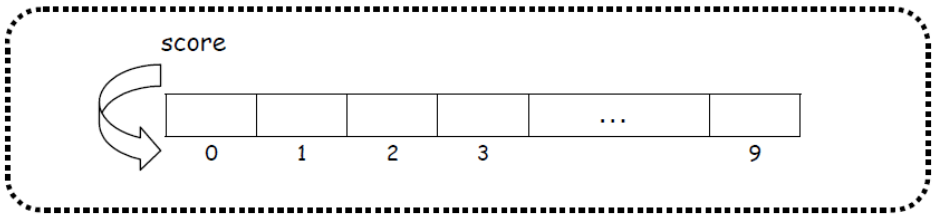
6.1 รูปแบบการประกาศตัวแปรชุด

ในภาษาซีเราสามารถสร้างตัวแปรชุดจากข้อมูลพื้นฐาน อันได้แก่ int float double เป็นต้น รูปแบบของการประกาศตัวแปรชุดทำได้ดังนี้

ชนิดข้อมูล ชื่อตัวแปร[ขนาดข้อมูล]

ตัวอย่าง 6.1 แสดงการประกาศตัวแปรชุดเพื่อเก็บข้อมูลคะแนนสอบของนักศึกษา 10 คน

```
float  score[10];
```



คำสั่ง float score[10]; เป็นการกำหนดตัวแปรตัวแปรชุดชื่อ score เป็นตัวแปรชุดชนิด float ที่มีสมาชิกทั้งหมด 10 ตัว ตั้งแต่ score[0], score[1], score[2], ... , score[9] **สมาชิกภายในตัวแปรชุดจะเริ่มที่ 0 เสมอ** และสมาชิกตัวสุดท้ายจะอยู่ที่ตำแหน่งของขนาดที่ประกาศไว้ลบด้วย 1 เช่น ประกาศขนาดของตัวแปรชุดไว้ n สมาชิกตัวสุดท้ายจะอยู่ที่ตำแหน่ง n-1

<u>การอ้างถึงสมาชิกแต่ละตัวภายในตัวแปรชุด</u>

การอ้างถึงสมาชิกของตัวแปรชุดจะใช้ระบบดัชนี (Index) โดยผ่านเครื่องหมาย [ ] เช่น อ้างถึงสมาชิกตำแหน่งแรกของตัวแปรชุดด้วย score[0] เป็นต้น การใช้งานสมาชิกของตัวแปรชุดสามารถใช้งานได้เหมือนตัวแปรพื้นฐานทั่วไป ตัวอย่างต่อไปนี้ แสดงคำสั่งที่ใช้งานกับสมาชิกของตัวแปรชุด

คำสั่งในการบวกค่าสมาชิก 3 ตัวแรกของตัวแปรชุด

```
sumThird = score[0] + score[1] + score[2];
```

คำสั่งในการกำหนดค่า 5 ให้กับสมาชิกตัวแรกของตัวแปรชุด

```
score[0] = 5;
```

คำสั่งในการเปรียบเทียบว่าค่าของสมาชิกตัวแรกมากกว่าสมาชิกตัวสุดท้ายหรือไม่

```
if ( score[0] > score[9] )
    printf (“First is greater than last\n” );
```

เราสามารถอ้างถึงสมาชิกทุกตัวภายในตัวแปรชุดอย่างอิสระ ภายในขอบเขตของขนาดที่ได้ประกาศตัวแปรชุดไว้ แต่การใช้ตัวแปรชุดทั่วไปจะเป็นการเข้าถึงสมาชิกโดยใช้ตัวแปรประเภท int มาช่วยเป็นดัชนีอ้างถึงสมาชิกที่ต้องการ

ตัวอย่างต่อไปนี้แสดงการเข้าถึงสมาชิกของตัวแปรชุดโดยใช้ตัวแปรประเภท int สมมติให้ i, j, k เป็นตัวแปรประเภท int

คำสั่งวนซ้ำ เพื่อพิมพ์ค่าที่เก็บอยู่ในแต่ละสมาชิกของตัวแปรชุด

```
for (k = 0; k < 10; k++)
printf (“Value at %d = %d\n”, k+1, score[k]);
```

คำสั่งกำหนดให้ค่าสมาชิกตำแหน่งที่ i+j หรือ 2+3 คือ สมาชิกตำแหน่งที่ 5 มีค่าเท่ากับ 0

```
i = 2;
j = 3;
score[i + j] = 0;
```

คำสั่งการรับค่าเพื่อนำ มาเก็บในตัวแปรตัวแปรชุดสามารถทำ ได้ด้วยคำสั่ง

```
for (i=0; i < 10; i++) {
        printf("Enter member %d : ", i);
        scanf("%f", &score[i]);
}
```

<u>หมายเหตุ</u>

- จากตัวอย่างจะเห็น ว่าการอ้างถึงสมาชิกแต่ละตัวภายในตัวแปรชุดจะใช้ลักษณะการอ้างถึงในลักษณะเดียวกับการอ้างถึงตัวแปรทั่วไป แต่เมื่อใดมีการอ้างถึงแต่ชื่อของตัวแปรตัวแปรชุด เช่น อ้างถึง score จะเป็นการอ้างถึงแอดเดรสเริ่มต้นของตัวแปรตัวแปรชุดนั้น

- ***สิ่งที่ต้องระวัง*** คือ ในภาษาซีจะไม่มีการกำหนดให้ตรวจสอบขอบเขตของตัวแปรชุด ผู้เขียนโปรแกรมจะต้องพยายามเขียนโปรแกรมที่เกี่ยวข้องกับสมาชิกของตัวแปรชุดภายในขอบเขตที่ประกาศตัวแปรชุด ไว้หากมีการอ้างอิงถึงสมาชิกตัวแปรชุดนอกขอบเขตที่ได้ระบุไว้ เช่น score[12] สิ่ง ที่ได้คือการไปอ่านข้อมูลในพื้นที่ของหน่วยความจำ ที่อาจจะเก็บค่าของตัวแปรตัวอื่นหรือเป็นค่าอื่นใดที่ไม่อาจคาดเดาได้

ตัวอย่างที่ 6.2 ให้รับค่าของจำนวนเต็ม 5 จำนวนจากผู้ใช้ และแสดงผลในลำดับที่กลับกัน

```
/* Read five integers from the standard input */
/* and output them in reverse order. */
# include <stdio.h>
#define SIZE 5
main ( ) {
    int k;                          /* loop control */
    int data[SIZE];                 /* data value */
    for (k = 0; k < SIZE; k++)
        scanf (“%d”, &data[k]);     /* data input */
    for (k = SIZE-1; k >= 0; k--)
        printf (“%d\n”, data[k]);   /* data output */
}
```

จากตัวอย่างจะเป็นการใช้ตัวระมวลผลก่อน (Preprocessor) #define กำหนดให้ SIZE มีค่าเป็น 5 ซึ่งในที่นี้เป็นการกำหนดค่าคงที่ให้กับขนาดของตัวแปรชุด กระบวนการทำ งานของตัวประมวลผลก่อนจะทำงานดังนี้ คือเมื่อสั่งให้มีการแปลคำสั่ง ตัวแปลคำสั่งจะแทนค่า 5 ลงไปทุกที่ที่เจอคำ ว่า SIZE แล้วจึงทำการคอมไพล์ต่อจนเสร็จเรียบร้อย

การทำงานของตัวอย่างข้างบนจะได้ตัวแปรชุดชื่อ data มีขนาด 5 หลังจากนั้นจะใช้คำสั่งวนซ้ำ for ให้อ่านค่าเข้าทางอุปกรณ์ป้อนข้อมูลมาตรฐาน (Standard Input ปกติจะเป็นคีย์บอร์ด) ด้วยฟังก์ชัน scanf ( ) มาเก็บยังสมาชิกของตัวแปรชุดแต่ละตัวตั้งแต่ตัวแรกคือ data[0] จนถึงตัวสุดท้ายคือ data[4] สังเกตว่าจะต้องส่งแอดเดรสของสมาชิกแต่ละตัวให้กับฟังก์ชัน scanf ( ) ด้วยเครื่องหมาย &

หลังจากนั้น นำข้อมูลที่ได้มาพิมพ์ออกทางอุปกรณ์แสดงผลมาตรฐาน (Standard Output ปกติจะเป็นจอภาพ) ในลำดับที่กลับกัน คือ จากตัวสุดท้ายคือ table[4] จนถึงตัวแรกคือ table[0]

สมาชิก ของตัวแปรชุดอาจเป็นประเภทข้อมูลใด ๆ ก็ได ้ เช่น

```
#define TSIZE 10
#define NAMESIZE 20
#define ADDRSIZE 30
int age[TSIZE];
float size[TSIZE+1];
char name[NAMESIZE], address[ADDRSIZE];
```

จากตัวอย่างจะได้ตัวแปรชุดชื่อ age เป็นประเภท int มีสมาชิก 10 ตัว ได้ตัวแปรชุดชื่อ size เป็น ประเภท float มีสมาชิก 11 ตัว ส่วนการประกาศข้อมูลสุดท้ายจะได้ตัวแปรชุดประเภท char คือ name มีสมาชิกเป็นตัวอักษร 20 ตัว และ address มีสมาชิก เป็นตัวอักษร 30 ตัว

ตัวอย่างที่ 6.3 เขียนโปรแกรมเพื่อรับข้อมูลจำนวนจริง 3 จำนวนจากผู้ใช้และหาค่าเฉลี่ยของค่าที่รับเข้ามาทั้งหมด ให้หาว่ามีเลขจำนวนจริงที่รับเข้ามานั้นมีค่าน้อยกว่าค่าเฉลี่ยกี่จำนวน โดยใช้ตัวแปรชุดในการเก็บข้อมูล

```
#include <stdio.h>
#define N 3
void readData(float [ ]);
void calAverage(float [ ], float *);
void findLessAverage(float [ ], float, int *);
void main( ) {
    float x[N], average;
    int num;
    readData(x);
    calAverage(x, &average);
    findLessAverage(x, average, &num);
    printf("Less than average = %d", num);
}
void readData(float px[ ]) {
    int i;
    for (i=0; i<N; i++) {
        printf("Enter value %d : ", i+1);
        scanf("%f", &px[i]);
    }
}
void calAverage(float a[ ], float *pAvg) {
    int i;
    float sum=0.0;
    for (i=0; i<N; i++) {
        sum = sum + a[i];
    }
    *pAvg = sum / N;
}
void findLessAverage(float b[ ], float avg, int *pNum) {
    int i;
    *pNum = 0;
    for (i=0; i<N; i++) {
        if (b[i] < avg)
            (*pNum)++;
    }
}
```

จากตัวอย่างที่ 6.3 สังเกตได้ว่าตัวแปรเลขจำนวนจริง 3 ตัว คือ x1, x2, x3 ฟังก์ชัน findLessAverage( ) จะต้องเขียนคำสั่งเปรียบเทียบเงื่อนไขที่ซ้ำกันทั้ง 3 ครั้ง การทำงานในลักษณะเช่นนี้นิยมใช้ตัวแปรชุดมาช่วย นอกจากนี้หากโจทย์เปลี่ยนใหม่เป็นให้รับข้อมูล 10 ตัว ในตัวอย่างที่ 6.3 สามารถแก้ไขได้ที่ตำแหน่งเดียวคือ เปลี่ยนคำ สั่ง #define เป็น

#define N 10

นอกจากนี้ให้สังเกตการส่งผ่านตัวแปรชุดไปยังฟังก์ชัน เช่น การรับข้อมูลเข้ามาเก็บในตัวแปรชุดของฟังก์ชัน readData( ) สามารถกำหนดโปรโตไทป์ได้ว่า

void readData(float [ ]) ;

จากโปรโตไทป์จะบอกให้ทราบว่ามีการส่งตัวแปรตัวแปรชุดชนิด float เข้าไปยังฟังก์ชัน การเรียกใช้งานจากฟังก์ชัน main( ) สามารถเรียกโดยอ้างถึงชื่อของตัวแปรชุดนั้นได้ทันที เช่น

```
float x[N];
readData(x);
```

สังเกตไม่จำเป็นต้องระบุขนาดของตัวแปรชุดในฟังก์ชัน การเปลี่ยนแปลงค่าใด ๆ ของตัวแปรตัวแปรชุดในฟังก์ชัน readData( ) จะเป็นการเปลี่ยนแปลงค่าของตัวแปรชุด x ในฟังก์ชัน main( ) ด้วย จากตัวอย่างตัวแปร px ในฟังก์ชัน readData( ) จะทำ หน้าที่คล้ายกับพอยน์เตอร์ชี้มายังตัวแปรชุด x ในฟังก์ชัน main( ) ซึ่งการอ้างถึงชื่อตัวแปรตัวแปรชุดจะเป็นการอ้างถึงแอดเดรสเริ่มต้นของตัวแปรชุดนั้น เพราะฉะนั้นเมื่อมีการรับข้อมูลมาเก็บยังตัวแปร px[i] ก็จะเหมือนกับการรับข้อมูลมาเก็บที่ตัวแปร x[i] นั่นเอง

แสดงตัวอย่างเพิ่มเติมดังตัวอย่างที่ 6.4 ถึง 6.6 ตัวอย่างที่ 6.4 แสดงการรับข้อมูลตัวแปรชุด 2 ตัวแปรชุดและนำ มาเปรียบเทียบกัน ตัวอย่างที่ 6.5 จะแสดงตัวอย่างการใช้ตัวแปรชุดในกรณีที่รับข้อมูลไม่เต็มจำนวนของสมาชิกตัวแปรชุดที่ได้จองไว ส่วนตัวอย่างที่ 6.6 แสดงตัวอย่างการใช้ตัวแปรชุดคู่ขนาน เป็นตัวแปรชุด 2 ตัวแปรชุด ที่เก็บข้อมูลสมาชิกของทั้ง 2 ตัวแปรชุดสอดคล้องกัน

ตัวอย่างที่ 6.4 โปรแกรมเพื่อรับข้อมูลตัวแปรชุดของจำนวนเต็ม 2 ตัวแปรชุด แต่ละตัวแปรชุดประกอบด้วยสมาชิก จำนวนเต็ม 5 จำนวน โดยมีเงื่อนไขการรับข้อมูลแต่ละตัวแปรชุด คือ สมาชิกตัวแรกเป็นเลขจำนวนเต็มใด ๆ ก็ได แต่ถ้าเป็นข้อมูลตัวอื่น ๆ ข้อมูลสมาชิกที่รับนั้นจะต้องมีค่ามากกว่าข้อมูลของสมาชิกก่อนหน้านั้น หลังจากรับข้อมูลแล้วให้นำ ข้อมูลจากตัวแปรชุดทั้ง 2 มาสร้างตัวแปรชุดผลลัพธ์ โดยเปรียบเทียบสมาชิกในตัวแปรชุดที่ 1 กับตัวแปรชุดที่ 2 ในลักษณะสมาชิกต่อสมาชิก หากสมาชิกในตัวแปรชุดใดมีค่ามากกว่าให้นำค่ามาเก็บในตัวแปรชุดที่ 3 ในตำแหน่งสมาชิกที่ตรงกัน กรณีที่มีค่าเท่ากันให้นำ สมาชิกในตัวแปรชุดใดมาก็ได ดังตัวอย่าง

ตัวแปรชุด 1	3	5	7	10	12
ตัวแปรชุด 2	2	5	9	10	11
ตัวแปรชุด 3	3	5	9	10	12

```
#include <stdio.h>
#define SIZE 5
void readArray (int [ ]);
void calArray (int [ ], int [ ], int [ ]);
void printArray (int [ ], int [ ], int [ ]);
void main() {
    int arr1[SIZE], arr2[SIZE], result[SIZE];
    printf("\nE N T E R F I R S T D A T A\n");
    readArray(arr1);
    printf("\nE N T E R S E C O N D D A T A\n");
    readArray(arr2);
    printf("\n\nC O M P A R E A R R A Y");
    calArray(arr1, arr2, result);
    printArray(arr1, arr2, result);
}
void readArray (int a[ ]) {
    int i=0, tmp;
    while (i < SIZE) {
        printf("Enter member #%d : ", i);
        scanf("%d", &tmp);
        if (i==0 || tmp > a[i-1])
            a[i++] = tmp;
        else
            printf("!!!ERROR!!!, data must more than %d\n", a[i-1]);
    }
}
void calArray (int first[], int second[], int ans[]) {
    int i;
    for (i=0; i < SIZE; i++)
        if (first[i] > second[i])
            ans[i] = first[i];
        else
            ans[i] = second[i];
    }
void printArray (int first[], int second[], int ans[]) {
    int i;
    printf("\n\nP R I N T R E S U L T");
    for (i=0; i < SIZE; i++)
        printf("\n%d. first [%d], second[%d] => ans [%d]", i, first[i], second[i], ans[i]);
}
```

ตัวอย่างที่ 6.5 เขียนโปรแกรมเพื่อรับข้อมูลคะแนนสอบของนักเรียนห้องหนึ่งซึ่งมีไม่เกิน 100 คน หากมีการป้อนข้อมูลคะแนน –999 แสดงว่าสิ้นสุดการป้อนข้อมูล ให้หาว่าคะแนนเฉลี่ยของการสอบครั้งนั้นเป็นเท่าใด และคะแนนที่สูงที่สุดและต่ำที่สุดเป็นเท่าใด

```
#include <stdio.h>
#define MAX 100
void readScore(float [ ], int *);
float findAvg(float [ ], int);
float findMax(float [ ], int);
float findMin(float [ ], int);
void main(){
    float score[MAX], avg, max, min;
    int num;
    avg = max = min = 0.0;
    readScore(score, &num);
```

```
    if (num > 0) {
        avg = findAvg(score, num);
        max = findMax(score, num);
        min = findMin(score, num);
    }
    printf("\nAverage is %.2f, Max. score is %.2f, Min. score is %.2f", avg, max, min);
}
void readScore(float pscore[], int *n) {
    int i=0;
    printf("Enter student no.%d score : ", i+1);
    scanf("%f", &pscore[i]);
    while (i < MAX && pscore[i] != -999) {
        i++;
        printf("Enter student no.%d score : ", i+1);
        scanf("%f", &pscore[i]);
    }
    *n = i;
}
float findAvg(float pscore[], int n) {
    int i;
    float sum=0.0, avg;
    for (i=0; i<n; i++)
        sum = sum + pscore[i];
    avg = sum / n;
    return(avg);
}
float findMax(float pscore[], int n) {
    int i;
    float max;
    max = pscore[0];
    for (i=1; i<n; i++) {
        if (max < pscore[i])
            max = pscore[i];
    }
        return(max);
}
float findMin(float pscore[], int n) {
    int i;
    float min;
    min = pscore[0];
    for (i=1; i<n; i++) {
        if (min > pscore[i])
            min = pscore[i];
    }
    return(min);
}
```

ในการใช้งานตัวแปรชุดของภาษาซีจะต้องมีการระบุขนาดของตัวแปรชุดที่แน่นอนในช่วงของการเขียนโปรแกรมทุกครั้ง แต่อาจจะใช้งานตัวแปรชุดไม่เต็มขนาดที่จองไว้ก็ได้ แต่ทั้งนี้ต้องคำนึงเรื่องการเสียเปล่าของหน่วยความจำ ที่ถูกจองพื้นที่ด้วย

ตัวอย่างที่ 6.6 เขียนโปรแกรมเพื่อรับข้อมูลอายุและความสูงของนักเรียนห้องหนึ่งซึ่งมีนักเรียน 30 คน ให้หาความสูงเฉลี่ยของนักเรียนในห้องนั้น และแสดงรายละเอียดอายุและความสูงของนักเรียนที่สูงกว่าความสูงเฉลี่ย และหาว่าอายุเฉลี่ยของนักเรียนที่สูงกว่าความสูงเฉลี่ยเป็นเท่าใด

```
#include <stdio.h>
#define MAX 30
void readStudentData(int [ ], float [ ]);
float findAverageHeight(float [ ]);
void printHigherAverage(float, int [ ], float [ ]);
void main() {
    int age[MAX];
    float height[MAX], avg;
    readStudentData(age, height);
    avg = findAverageHeight(height);
    printHigherAverage(avg, age, height);
}
void readStudentData(int pAge[ ], float pHeight[ ]) {
    int i;
    for (i=0; i<MAX; i++) {
        printf("Enter data for student no.%d\n", i+1);
        printf("Enter age : ");
        scanf("%d", &pAge[i]);
        printf("Enter height (cm.) : ");
        scanf("%f", &pHeight[i]);
    }
}
float findAverageHeight(float pHeight[ ]) {
    int i;
    float sum=0.0, average;
    for (i=0; i<MAX; i++) {
        sum += pHeight[i];
    }
    average = sum / MAX;
   return(average);
}
void printHigherAverage(float avgHeight, int pAge[ ], float pHeight[ ]) {
    int i, count=0;
    float sum=0.0, avgAge;
    printf("\n\nHigher than average height %.2f cm.", avgHeight);
    for (i=0; i<MAX; i++) {
        if (pHeight[i] > avgHeight) {
            printf("\nStudent no.%-2d, age %d years old, height %.2f cm.",i+1, pAge[i], pHeight[i]);
            sum += pAge[i];
            count++;
        }
     }
    avgAge = sum / count;
    printf("\n\nAverage age in this group is %.2f", avgAge);
}
```

การกำหนดค่าเริ่มต้นให้กับตัวแปรชุด สามารถทำได้ในช่วงเริ่มต้นของการประกาศตัวแปร เช่น ต้องการกำหนดราคาสินค้าภายในร้านค้าแห่งหนึ่ง โดยเก็บข้อมูลเป็นตัวแปรชุด สามารถทำได้ด้วยคำ สั่ง

```
float price[ ] = {100.0, 120.0, 85.0, 90.0, 150.0};
```

ข้อมูลภายในเครื่องหมายปีกกาจะถูกกำหนดให้เก็บในตัวแปรชุดชื่อ price ในตำแหน่งสมาชิกเริ่มต้นจาก 0

จนถึงสมาชิกตัวสุดท้าย และสังเกตว่าไม่มีการประกาศขนาดของตัวแปรชุด แต่ขนาดของตัวแปรชุดจะขึ้นอยู่กับค่าที่กำหนดให้ว่ามีจำนวนกี่ตัว ในที่นี้จะได้ว่าตัวแปรชุด price มีสมาชิก 5 ตัว แสดงได้ดังรูปที่ 6.1 และแสดงการใช้งานดังตัวอย่างที่ 6.7

price	100.0	120.0	85.0	90.0	150.0
	price[0]	price[1]	price[2]	price[3]	price[4]

รูปที่ 6.1 แสดงข้อมูลที่เก็บในตัวแปรชุด price

ตัวอย่างที่ 6.7 เขียนโปรแกรมเพื่อคำนวณปริมาตรน้ำมันที่ลูกค้าจะได้รับเมื่อเติมน้ำมันที่สถานีบริการ น้ำมันแห่งหนึ่ง โดยมีประเภทน้ำมันอยู่ 3 ประเภทและเก็บราคาอยู่ในตัวแปรชุดดังนี้

สมาชิกที่ 0 เก็บราคาน้ำมันเบนซิน 91 มีค่า 14.50 บาท

สมาชิกที่ 1 เก็บราคาน้ำมันเบนซิน 95 มีค่า 15.50 บาท

สมาชิกที่ 2 เก็บราคาน้ำมันดีเซล มีค่า 12.50 บาท

```
#include <stdio.h>
#define MAX 3
void readData(int *, float *, float [ ]);
float calVolume(int, float, float [ ]);
void main() {
    int type;
    float volume, amount, price[ ]={14.5, 15.5, 12.5};
    readData(&type, &amount, price);
    volume = calVolume(type, amount, price);
    printf("\n\t\t\tYou will got %.2f litres", volume);
}
void readData(int *t, float *a, float price[ ]) {
    int i;
    printf("\n\t\t\tFuel price");
    for (i=0; i<MAX; i++) {
        printf("\n\t\t%d. ", i+1);
        switch ( i ) {
            case 0 : printf("%-15s -> ", "Benzene 91");
                break;
            case 1 : printf("%-15s -> ", "Benzene 95");
                break;
            default : printf("%-15s -> ", "Diesel");
    }
    printf("%.2f Baht/litre", price[i]);
    }
    do {
        printf("\n\n\t\t\tSelect type (1-3) : ");
        scanf("%d", t);
    } while (!(*t >= 1 && *t <= 3));
    do {
        printf("\n\t\t\tAmount (Baht) : ");
       scanf("%f", a);
    } while (!(*a > 0));
}
```

```
float calVolume(int type, float amount, float price[]) {
    int vol;
    vol = amount / price[type-1];
    return(vol);
}
```

6.1 การใช้พอยน์เตอร์กับตัวแปรชุด

การทำงานใด ๆ ของตัวแปรชุดสามารถใช้พอยน์เตอร์เข้ามาช่วย ซึ่งจะทำให้มีความเร็วในการทำงานสูงขึ้น สมมติว่ามีตัวแปรชุด a และพอยน์เตอร์ pa ดังนี้

int a[10];

*int *pa;*

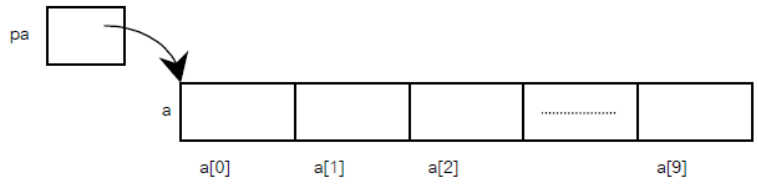
เนื่องจากการอ้างถึง ชื่อของตัวแปรชุดจะเป็นการอ้างถึงแอดเดรสเริ่มต้นของตัวแปรชุดนั้น หากต้องการให้ตัวแปรพอยน์เตอร์ pa มาชี้ยังตัวแปรชุด a สามารถทำ ได้โดยใช้คำสั่ง

pa = a;

ซึ่งมีการทำงานจะเหมือนกับการใช้คำ สั่ง

pa = &a[0];

pa จะเก็บ ค่าแอดเดรสเริ่มต้นของตัวแปรชุด a แสดงดังรูปที่ 6.2



รูปที่ 6.2 แสดงพอยน์เตอร์ชี้ไปยังแอดเดรสเริ่มต้นของตัวแปรชุด

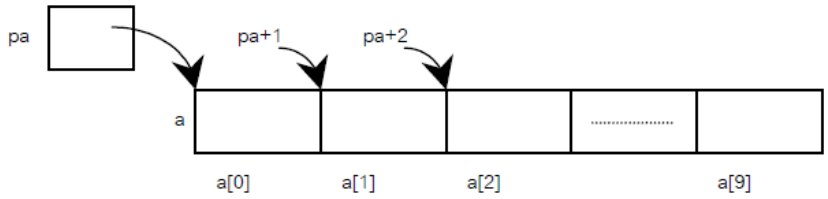
การนำพอยน์เตอร์นั้นไปใช้งานสามารถทำ ได้โดยอ่านค่าตัวแปรชุดผ่านพอยน์เตอร์ดังนี้

*int x = *pa;*

จะเป็นการกำหนดค่าให้ x มีค่าเท่ากับ a[0] การเลื่อนไปอ่านค่าสมาชิกตำแหน่งต่าง ๆ ของตัวแปรชุดผ่านทางพอยน์เตอร์สามารถทำได้โดยการเพิ่มค่าพอยน์เตอร์ขึ้น 1 เพื่อเลื่อนไปยังตำแหน่งถัดไป หรือเพิ่มค่าขึ้น N เพื่อเลื่อนไป N ตำแหน่ง หรืออาจจะลดค่าเพื่อเลื่อนตำแหน่งลง กรณีที่ pa ชี้อยู่ที่ a[0] คำสั่ง

pa+1;

จะเป็นการอ้างถึงแอดเดรสของ a[1] หากเป็น pa+i เป็นการอ้างถึงแอดเดรส a[i] หากต้องการอ้างถึงข้อมูล ภายในของสมาชิกของตัวแปรชุดตำแหน่งที่ a[i] จะใช้ *(pa+i) แสดงดังรูปที่ 6.3



รูปที่ 6.3 แสดงการอ้างถึงตำแหน่งในตัวแปรชุดผ่านพอยน์เตอร์

การสั่งให้บวก 1 หรือ บวก i หรือ ลบ i เป็นเหมือนการเลื่อนไปยังสมาชิกของตัวแปรชุดตำแหน่งที่ต้องการ เนื่องจากประเภทของข้อมูลแต่ละประเภทของตัวแปรชุด เช่น int, float, float และอื่น ๆ มีขนาดของข้อมูลที่ต่างกัน ทำให้ขนาดของสมาชิกภายในตัวแปรชุดแต่ละประเภทมีขนาดแตกต่างกันด้วย การสั่งให้บวกหรือลบด้วยจำนวนที่ต้องการนั้นจะมีกลไกทำหน้าที่คำนวณตำแหน่งที่ต้องการให้สอดคล้องกับข้อมูลแต่ละประเภทโดยอัตโนมัติ

นอกจากนี้ยังสามารถใช้พอยเตอร์แทนตัวแปรชุด การอ้างโดยใช้ a[i] สามารถใช้ *(a+i) เนื่องจากทุกครั้งทที่อ้างถึง a[i] ภาษาซีจะทำ หน้าที่แปลงเป็น *(a+i) เพราะฉะนั้นการเขียนในรูปแบบใดก็ให้ผลลัพธ์การทำงานเช่นเดียวกัน และการอ้างถึงแอดเดรส เช่น &a[i] จะมีผลเท่ากับการใช้ a+i

ในลักษณะเดียวกัน การใช้งานพอยน์เตอร์ก็สามารถใช้คำสั่งในลักษณะตัวแปรชุดก็ได เช่น การอ้างถึง *(pa+i) สามารถเขียนด้วย pa[i] ก็ได้ผลเช่นเดียวกัน

สิ่งที่แตกต่างกันของตัวแปรชุดและพอยน์เตอร์ คือ พอยนเตอร์เป็นตัวแปร แต่ตัวแปรชุดไม่ใช่ตัวแปรสมมติให้ a เป็นตัวแปรชุดและ pa เป็นพอยน์เตอร์ การอ้างถึง pa = a หรือ pa++ จะสามารถคอมไพล์ได้ แต่จะไม่สามารถใช้คำ สั่ง a = pa หรือ a++ ได้ เมื่อมีการส่งชื่อของตัวแปรชุดให้แก่ฟังก์ชัน จะเป็นการส่งตำแหน่งแอดเดรสของสมาชิกตัวแรกของตัวแปรชุดให้แก่ฟังก์ชัน ดังนั้นพารามิเตอร์ในฟังก์ชันนั้นจะเป็นตัวแปรประเภทพอยน์เตอร์ แสดงดังตัวอย่างที่ 6.8

ตัวอย่างที่ 6.8 เป็นการแปลงโปรแกรมในตัวอย่างที่ 6.6 ให้เป็นการเขียนในลักษณะพอยน์เตอรทั้งหมดซึ่ง ให้ผลลัพธ์การทำงานที่เหมือนกัน

```
#include <stdio.h>
#define MAX 5
void readStudentData(int *, float *);
float findAverageHeight(float *);
void printHigherAverage(float, int *, float *);
void main() {
    int age[MAX];
    float height[MAX], avg;
    readStudentData(age, height);
    avg = findAverageHeight(height);
    printHigherAverage(avg, age, height);
}
void readStudentData(int *pAge, float *pHeight) {
    int i;
    for (i=0; i<MAX; i++) {
        printf("Enter data for student no.%d\n", i+1);
        printf("Enter age : ");
        scanf("%d", pAge+i);
        printf("Enter height (cm.) : ");
        scanf("%f", pHeight+i);
    }
}
```

```
float findAverageHeight(float *pHeight) {
    int i;
    float sum=0.0, average;
    for (i=0; i<MAX; i++) {
        sum += *(pHeight+i);
    }
    average = sum / MAX;
    return(average);
}
void printHigherAverage(float avgHeight, int *pAge, float *pHeight) {
    int i, count=0;
    float sum=0.0, avgAge;
    printf("\n\nHigher than average height %.2f cm.", avgHeight);
    for (i=0; i<MAX; i++) {
        if (*(pHeight+i) > avgHeight) {
            printf("\nStudent no.%-2d, age %d years old, height %.2f cm.",
            i+1, *(pAge+i), *(pHeight+i));
            sum += *(pAge+i);
            count++;
        }
    }
    avgAge = sum / count;
    printf("\n\nAverage age in this group is %.2f", avgAge);
}
```

6.3 ตัวแปรชุดของตัวอักขระ

ในกรณีของการใช้ตัวแปรชุดกับข้อมูลชนิด char จะมองเป็นการทำงานกับข้อมูลหลาย ๆ ตัวอักขระหรือข้อความที่เรียกว่าสตริง (String) ค่าคงที่ของข้อความเป็นสิ่งที่มีการใช้เสมอในโปรแกรม เช่น “Hello” แต่ในการใช้งานตัวแปรสตริงจะต้องมีการเตรียมพื้นที่ในการเก็บข้อความเผื่อไว้หนึ่งตำแหน่งเสมอ เช่น หากต้องการประกาศตัวแปรเพื่อเก็บข้อความว่า “Mickey Mouse” จะต้องจองพื้นที่เท่ากับจำนวนตัวอักขระที่มีและบวกไปด้วย 1 เสมอ เนื่องจากลักษณะการเก็บข้อมูลประเภทข้อความในหน่วยความจำ จะมีการปะตัวอักษร Null หรือ ‘\0’ ต่อท้ายเสมอเพื่อให้รู้ว่าเป็นจุดสิ้นสุดของข้อมูล ในที่นี้ต้องจองพื้นที่ขนาด 13 ตัวอักขระ การจองพื้นที่ดังกล่าวจะเหมือนการจองพื้นที่ของข้อมูลประเภทตัวแปรชุดเป็นตัวแปรชุดของ char สามารถใช้คำ สั่งในการประกาศตัวแปรคือ

char message[13];

หากต้องการกำหนดค่าให้กับตัวแปร message สามารถทำ ได้ด้วยคำ สั่ง

char message[ ] = “Mickey Mouse”;

จำลองการเก็บข้อมูลดังกล่าวในหน่วยความจำ ดังรูปที่ 6.4

M	i	c	k	e	y		M	o	u	s	e	\0

รูปที่ 6.4 แสดงแบบจำลองการเก็บข้อมูลประเภทสตริงในหน่วยความจำ

ค่าคงที่สตริงที่พบเห็นได้เสมอได้แก่ข้อความที่ใช้ในฟังก์ชัน printf( ) เช่น

printf (“Hello, world\n”);

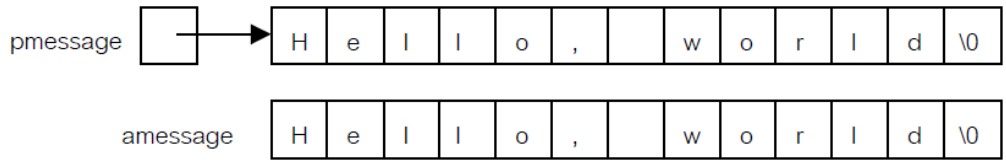
ฟังก์ชัน printf( ) จะรับพารามิเตอร์เป็นพอยน์เตอร์ชี้ไปยังแอดเดรสของข้อมูลที่ตำแหน่งเริ่มต้นของตัวแปรชุด และนำ ข้อความนั้นแสดงออกทางอุปกรณ์แสดงข้อมูลมาตรฐาน ในการเขียนโปรแกรมจะสามารถใช้พอยน์เตอร์ชี้ไปค่าคงที่สตริงใด ๆ ก็ได้ เช่น

*char *pmessage = “Hello, world”;*

pmessage จะเป็นพอยเตอร์ประเภท char ชี้ไปที่ตัวแปรชุดของตัวอักษร จะแตกต่างจากการใช้ตัวแปรชุดทั่วไปเช่น

char amessage[ ] = “Hello, world”;

ลักษณะของตัวแปรชุดเช่น amessage จะมีการจองพื้นที่ใช้กับตัวแปรชุดขนาด 13 ตัวอักษรรวมทั้ง Null ส่วนลักษณะของพอยน์เตอร์ชี้ไปยังค่าคงที่สตริง จะมีการจองพื้นที่ให้กับค่าคงที่สตริงขนาด 13 ตัวอักษรเช่นเดียวกันแต่จะมีการจองพื้นที่ให้กับพอยน์เตอร์และทำการชี้พอยน์เตอร์นั้นไปยังพื้นที่ของค่าคงที่สตริงที่จองเอาไว แสดงดังรูปที่ 6.5 และแสดงตัวอย่างการใช้งานดังตัวอย่างที่ 6.9



รูปที่ 6.5 แสดงการจองพื้นที่ให้กับตัวแปรชุดและพอยน์เตอร์ชี้ไปยังค่าคงที่สตริง

ตัวอย่างที่ 6.9 ฟังก์ชันที่รับพารามิเตอร์เป็นพอยน์เตอร์ โดยอาร์กิวเมนท์ที่ส่งมาเป็นตัวแปรชุด เป็นการหาความยาวข้อความที่ส่งเข้ามายังฟังก์ชัน

```
#include <stdio.h>
int strlen(char *s) {
    int n;
    for (n=0; *s!='\0'; s++)
        n++;
    return n;
}
void main() {
    char str[ ]="I love C";
    int len;
    len = strlen(str);
    printf("Length of %s is %d", str, len);
}
```

จะเห็นว่า s เป็นพอยน์เตอร์ ในฟังก์ชันจะมีการตรวจสอบข้อมูลว่ามีค่าเท่ากับ ‘\0’ หรือไม่ และมีการเลื่อนตำแหน่งทีละ 1 ค่า (นับว่าข้อูลมีความยาวเพิ่มขึ้นทีละ1) โดยใช้ s++ คำตอบที่ได้จากการเรียกใช้ฟังก์ชันนี้จะเป็นความยาวของข้อมูลที่เก็บอยู่ภายในโดยที่ไม่รวมค่า Null การเรียกใช้ฟังก์ชัน strlen สามารถทำ ได้หลายลักษณะ

strlen ("Hello world"); */* string constant */*

strlen (array); */* char array[ ]="Hello world"; */*

strlen (ptr); */* char *ptr="Hello world"; */*

นอกจากนี้ยังอาจจะประกาศพารามิเตอร์ภายในฟังก์ชัน strlen ได้ใน 2 ลักษณะ คือ char *s แบบในตัวอย่าง หรืออาจจะใช้ char s[ ] ก็ได้ โดยทั่วไปจะใช้ในลักษณะแรก เพราะช่วยให้รู้ได้ทันทีว่า s เป็นตัวแปรพอยน์เตอร์ และยังสามารถส่งส่วนใดส่วนของตัวแปรชุดให้แก่ฟังก์ชันก็ได้ โดยไม่จำเป็นต้องส่งสมาชิกตัวแรกก็ได้เช่นกัน เช่น

strlen(&str[2]) หรือ strlen(str+2)

เป็นการส่งแอดเดรสของสมาชิก str[2] ให้กับฟังก์ชัน strlen( ) การประกาศฟังก์ชัน strlen( ) สามารถทำได้โดยการประกาศ

int strlen(char s[ ]) { …… } หรือ int strlen(char *arr) { …….. }

แสดงตัวอย่างเพิ่มเติม ของการทำงานของสตริงดังตัวอย่างที่ 6.10 ถึง 6.12 เป็นการใช้ฟังก์ชัน strcpy( ) ที่ทำหน้าที่ในการทำสำเนา (Copy) ข้อความจากตัวแปรหนึ่งไปยังอีกตัวแปร สามารถเขียนโดยในพารามิเตอร์ในลักษณะตัวแปรชุดและพอยน์เตอร์ได้ดังตัวอย่าง

ตัวอย่างที่ 6.10 ฟังก์ชัน strcpy ( ) ทำหน้าที่สำเนาข้อความจากตัวแปรหนึ่งไปยังอีกตัวแปรหนึ่งเขียนในลักษณะตัวแปรชุด

```
void strpy ( char *s , char *t ) {
    int i=0;
    while ( ( s[i] = t[i] ) != '\0' )
        i++;
}
```