บทที่ 6
โครงสร้างข้อมูล : อาเรย์ และ ระเบียน

วัตถุประสงค์

1. เพื่อให้นักศึกษาทราบถึงรูปแบบคำสั่งในการกำหนดข้อมูลชนิดอาเรย์
2. เพื่อให้นักศึกษาสามารถเข้าถึงข้อมูลเป็นลำดับในโครงสร้างข้อมูลชนิดในอาเรย์ได้
3. เพื่อให้นักศึกษาทราบ เข้าใจ และสามารถส่งผ่านอาร์กิวเมนต์ที่เป็นข้อมูลชนิดอาเรย์ได้
4. เพื่อให้นักศึกษาสามารถค้นหา และเรียงลำดับข้อมูลในโครงสร้างข้อมูลชนิดอาเรย์ได้
5. เพื่อให้นักศึกษาสามารถแก้ปัญหาโดยใช้โครงสร้างข้อมูลชนิดอาเรย์ได้
6. เพื่อให้นักศึกษาทราบรูปแบบคำสั่งในการกำหนดข้อมูลชนิด struct
7. เพื่อให้นักศึกษาสามารถแก้ปัญหาโปรแกรมโดยใช้ข้อมูลชนิด struct ได้

จากบทที่ผ่านมานั้นเป็นการแก้ปัญหาโปรแกรมที่มีการประกาศตัวแปรในการเก็บค่าใดๆได้เพียง 1 ค่าเท่านั้น ในกรณีที่ต้องการเก็บข้อมูลจำนวนมากต้องประกาศตัวแปรหลายตัวเท่ากับจำนวนข้อมูล ในกรณีที่เกิดปัญหาหรือนำข้อมูลต่างๆที่ต้องการมาใช้งาน โปรแกรมเมอร์ต้องทราบที่อยู่ของข้อมูลเหล่านั้นถ้าลืมหรือกำหนดชื่อไม่สื่อความอาจทำให้ยุ่งยากในการแก้ไข ภาษา C++ได้จัดเตรียมโครงสร้างข้อมูลหลายๆชนิดขึ้นมาเพื่อให้โปรแกรมเมอร์ได้กำหนดหรือเก็บข้อมูลที่ต้องการให้อยู่ในรูปแบบที่ง่ายต่อการนำมาใช้งาน สำหรับในบทนี้จะกล่าวถึงโครงสร้างข้อมูลชนิดอาเรย์ และ struct

6.1 ข้อมูลชนิดอาเรย์

เป็นโครงสร้างข้อมูลที่ยินยอมให้ตัวแปร 1 ตัวสามารถเก็บข้อมูลได้หลายค่าขึ้นอยู่กับความต้องการ อาทิเช่นต้องการเก็บข้อมูลคะแนนสอบนักศึกษาจำนวน 100 คน เราสามารถประกาศตัวแปร score ให้สามารถเก็บคะแนนของนักศึกษาจำนวน 100 คนนี้ได้โดยนำข้อมูลมาเรียงต่อๆกันเป็นลำดับ เริ่มจากลำดับที่ 0 , 1 , ... , 99 โดยข้อมูลคะแนนสอบของนักศึกษาคนแรกจะเก็บอยู่ในตัวแปร score โดยสามารถอ้างได้ดังนี้ score[0] ดังนั้น คะแนนของคนที่สองเก็บอยู่ใน score[1] ดังนั้นคนสุดท้ายคะแนนจะเก็บอยู่ใน score[99]

การประกาศตัวแปร

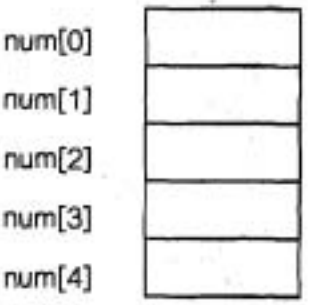
ตัวแปรชนิดอาเรย์นั้นจะมีการนำข้อมูลที่มีชนิดเดียวกันมาเก็บรวมกัน ซึ่งการอ้างข้อมูลต่างๆที่เก็บในอาเรย์นั้นอาจเป็นชนิดหนึ่งมิติ (one-dimension) สองมิติหรือหลายมิติ(two-more dimension) สำหรับในหัวข้อนี้จะกล่าวเฉพาะอาเรย์หนึ่งมิติเท่านั้น

รูปแบบของการประกาศตัวแปรหนึ่งมิติ

```
dataType arrayName[intExp] ;
```

เช่น int num[5] ; การประกาศตัวแปรนี้จะส่งผลให้ตัวแปร num สามารถเก็บข้อมูลได้ 5 จำนวนเป็นเลขจำนวนเต็ม อันประกอบด้วยตัวแปร num[0] ,num[1] ,num[2] ,num[3] ,num[4]

โดยโปรแกรมจะทำการจองเนื้อที่ว่างๆเป็นพื้นเดียวกันในหน่วยความจำดังนี้



Array Declaration	
รูปแบบ:	element-type array-name[size] ;
	element-type array-name[size] = {initialization-list} ;
ตัวอย่าง:	char myName[5] ;
	float salaries[NUM_EMP] ;
	char vowels[] = {'A' ,'E','I','O','U'} ;

Array Subscript	
รูปแบบ:	name[subscript] ;
ตัวอย่าง:	NUM[3*I-2]

เราสามารถปฏิบัติงานกับตัวแปรเหล่านี้ได้ เช่น

```
num[3] = 5 ;
cin >> num[4] ;
cout << num[6] ;
sum = num[3] + num[2] ;
sum += num[1] ;
```

เราสามารถนำค่าของข้อมูลที่เก็บในตัวแปรชนิดอาเรย์มาใช้งานได้เพียงแต่ระบุชื่อตัวแปรและลำดับ หรือตัวชี้ (index) ข้อมูลให้ถูกต้อง เรานิยมใช้คำสั่ง for มาใช้ในกำหนดหรือเก็บข้อมูลหรือการเข้าถึงข้อมูลในตัวแปรอาเรย์ในลำดับต่างๆ โดยให้ตัวแปรที่เป็นตัวนับเริ่มจาก 0 จนถึงลำดับสุดท้าย ดังนี้

1. กรณีที่กำหนดค่าเริ่มต้นให้แก่ตัวแปรอาเรย์

```
for (index=0 ; index<10 ; index++)
    num[index] = 0 ;
```

หรือกำหนดค่าพร้อมกับประกาศตัวแปรชนิดอาเรย์ได้ดังนี้

```
int num[10] = {0} ;
```

2. กรณีที่อ่านข้อมูลมาเก็บในตัวแปรอาเรย์

```
for (index=0 ; index<10 ; index++)
    cin >> num[index] ;
```

3. กรณีพิมพ์ข้อมูลที่เก็บในตัวแปรอาเรย์

```
for (index=0 ; index<10 ; index++)
    cout << num[index] << "  " ;
```

4. หาผลรวมและค่าเฉลี่ยของข้อมูลที่เก็บในตัวแปรอาเรย์

```
sum = 0 ;
for (index = 0 ; index<10 ; index++)
    sum = sum + num[index] ;
average = sum /10.0 ;
```

5. หาค่าที่มากที่สุดของข้อมูลที่เก็บในอาเรย์

```
maxIndex = 0;
for (index = 1; index<10 ; index++)
    if (num[maxIndex] < num[index])
        maxIndex = index ;
largestSale = num[maxIndex] ;
```

จะเห็นได้ว่าการใช้งาน หรือการเข้าถึงข้อมูลนั้นกระทำได้ง่าย ในกรณีที่ข้อมูลมีจำนวนมากขึ้นเราสามารถปรับเปลี่ยนคำสั่งโปรแกรมเพียงนิดหน่อย ไม่ต้องเขียนโปรแกรมใหม่ อาทิเช่นต้องการกระทำงานกับกลุ่มข้อมูล 100 เพียงเปลี่ยนค่าสุดท้ายของการทำงานจาก 10 เป็น 100 เท่านั้นเอง

ตัวอย่างที่ 6.1 โปรแกรมหาความแตกต่าง

เป็นโปรแกรมที่คำนวณหาค่าเฉลี่ยของกลุ่มตัวเลขและหาความแตกต่างระหว่างค่าของข้อมูลกับค่าเฉลี่ยของข้อมูล พิมพ์ออกมาทางจอภาพ

```
// File: showDiff.cpp
// Computes the average value of an array of data and displays the difference
//      between each value and the average
#include <iostream>
using namespace std;
int main()
{
    const int MAX_ITEMS = 8;
    float x[MAX_ITEMS] ;                    // array of data
    float average;                          // average value of data
    float sum ;                             // sum of the data
    // Enter the data.
     cout << "Enter " << MAX_ITEMS << " numbers: " ;
```

```
    for (int I = 0 ; I < MAX_ITEMS ; I++)
       cin >> x[I] ;
    // Compute the average value.
    sum = 0.0 ;                                  // initialize sum
    for (int I = 0 ; I<MAX_ITEMS ; I++)
       sum  += x[I] ;                            // add next element to sum
    average = sum / MAX_ITEMS ;                  // get average value
    cout << "The average value is " << average << endl << endl ;

    // Display the difference between each item and the average.
    cout << "Table of difference between x[I] and the average. " << endl ;
    cout << "   " << "I" << "   " << "x[I]" << "  " << "difference" << endl;
    for ( int I = 0 ; I< MAX_ITEMS ; I++)
       cout << "  " << I << "  " << x[I] << "  " << (x[I]-average) << endl ;
    return 0 ;
}
```

ทดสอบ

Enter 8 numbers: 16 12 6 8 2.5 12 14 -54.5

The average value is 2

Table of differences between x[I] and the average.

I	x[I]	difference
0	16	14
1	12	10
2	6	4
3	8	6
4	2.5	0.5
5	12	10

6 14 12

7 -54.5 -56.5

โปรแกรมนี้เป็นโปรแกรมในการนำเลขจำนวนจริงใดๆ 8 จำนวนเก็บในตัวแปรอาเรย์ x มีการใช้คำสั่ง for เพื่ออ้างถึงข้อมูลโดยให้เริ่มต้นที่ข้อมูลตัวแรกเพื่อนำเข้าข้อมูลใดๆที่ผู้ใช้ป้อนนำมาเก็บเรียงต่อๆกันทั้งสิ้น 8 จำนวน ต่อจากนั้นดึงค่าที่เก็บมาเก็บสะสมค่าในตัวแปร sum และหาค่าเฉลี่ยเก็บในตัวแปร average ต่อจากนั้นนำมาหาผลต่างระหว่างค่าที่เก็บกับค่าเฉลี่ย พิมพ์ออกมาทางจอภาพ

ตัวอย่างที่ 6.2 Cryptogram

โปรแกรมนี้เป็นการแปลงข้อมูลโดยผู้ใช้ป้อนข้อความใดๆทางแป้นพิมพ์ โปรแกรมจะนำอักขระในลำดับถัดไปมาแทนที่ข้อมูลในข้อความนั้น เช่น นำอักขระ 'b' แทนที่ 'a' , นำอักขระ 'c' แทนที่ 'b' สำหรับ 'a' แทนที่ 'z' เป็นต้น

```
// File :cryptogram.cpp
// Display a cryptogram
#include <string>
#include <iostream>
#include <cctype>
using namespace std;

int main()
{
   const string ALPHABET = "abcdefghijklmnopqrstuvwxyz" ;
   const string CODE =     "bcdefghijklmnopqrstuvwxyza" ;
   string message ;                          // message to endcode
   char  ch ;                                // next message character
   int pos ;                                 // its position
   cout << "Enter a string to encode: " ;
```

```
    cin >> message ;

    // Encode message.
    for (int I = 0 ; I < message.length() ; I++)
    {
        ch = tolower(message.at(I));              // ch to lowercase
        pos = ALPHABET.find(ch) ;                 // find position of ch
        if ((pos >= 0) && (pos<26))
            message[I] = CODE.at(pos) ;
    }
    cout << "The cryptogram is          :" << message << endl ;

    return 0 ;
}
```

ทดสอบ

```
Enter a string to encode:  This is a string.
The cryptogram is          : uijt jt b tusjoh
```

โปรแกรมนี้จะทำการประมวลผลโดยแปลงตัวอักษรใดๆให้กลายเป็นรหัสตัวอักษรในลำดับถัดไป การแก้ปัญหานั้นเราจะนำตัวอักษรภาษาอังกฤษทุกตัวเก็บในตัวแปรอาเรย์ ALPHABET และเก็บตัวอักษรที่ขยับตำแหน่งไปหนึ่งลำดับในตัวแปรอาเรย์ CODE เพื่อใช้เป็นการอ้างอิงในการค้นหาและปรับเปลี่ยน ซึ่งวิธีการนี้ทำให้ง่ายและสะดวกกว่าการใช้เงื่อนไขเพื่อตรวจสอบทางเลือกซึ่งมีมากถึง 26 ทางเลือก ทำให้โปรแกรมมีความซับซ้อนมากเกินความจำเป็น

ตัวอย่างที่ 6.3 โปรแกรมตัดเกรดนักศึกษา

โปรแกรมต่อไปนี้เป็นโปรแกรมในการรับคะแนนของนักศึกษาจำนวน N คนทางแป้นพิมพ์ เก็บในตัวแปรอาเรย์ชื่อ STU ต่อจากนั้นอ่านข้อมูลจากตัวแปรอาเรย์ STU เพื่อหาค่าคะแนนที่มากที่สุด คะแนนเฉลี่ย และ จำนวนของนักศึกษาที่สอบได้เกรด P ,G และ F โดยใช้กฎเกณฑ์ของมหาวิทยาลัยรามคำแหง สำหรับตัวอย่างนี้จะเขียนโปรแกรมโดยมีฟังก์ชัน main เพียงฟังก์ชันเดียว เพื่อเปรียบเทียบกับการออกแบบและเขียนโปรแกรมโดยแบ่งโปรแกรมนี้ออกเป็นโมดูลย่อยๆ เพื่อให้นักศึกษาเห็นถึงความแตกต่างของการออกแบบ ในหัวข้อถัดไป

```
#include <iostream>
using namespace std;
int main()
{
    int   STU[1000] ;                    // เก็บคะแนนของนักศึกษาสูงสุด1000 คน
    int N , I ;                          // จำนวนของนักศึกษา
    int  MAX ;                           // คะแนนที่มากที่สุด
    int  F=0 , P=0 , F=0 ;               // จำนวนของนักศึกษาที่ได้เกรดต่างๆ
    float  AVG ;                         // คะแนนเฉลี่ย
    float  SUM ;                         // ผลรวมของคะแนนทั้งหมด
    cout << "จำนวนของนักศึกษาทั้งหมด : " ;  cin >> N ;
    for (I=0 ; I< N ;I++)
    {
        cout << " คะแนนของนักศึกษาคนที่ " << I+1 << "=" ;
        cin >> STU[I] ;
    }
    // หาค่าที่มากที่สุดของคะแนนสอบ
    MAX = STU[0] ;
    for (I=1 ; I<N ; I++)
    {
```

```
        if (MAX < STU[I])
            MAX = STU[I] ;
    }
    //หาค่าเฉลี่ย
    SUM = 0.0 ;
    for(I=0 ; I<N ; I ++)
        SUM = SUM + STU[I];
    AVG = SUM / N ;
    // ตัดเกรดเพื่อหาจำนวนของนักศึกษาที่ได้เกรดต่างๆ
   for ( I=0 ; I<N ; I++)
    {
       if (STU[I] <60)
                F++ ;
       else if  (STU[I] < 80 )
                P++;
       else
                G++ ;
    }
    //พิมพ์ผลลัพธ์ออกทางจอภาพ
    cout << "จำนวนของนักศึกษาที่ลงทะเบียนทั้งหมด = " << N << "คน" << endl ;
    cout << "คะแนนสูงสุดคือ " << MAX << endl;
    cout << " คะแนนเฉลี่ยคือ " << AVG << endl;
    cout << " เกรด F = " << F << " คน" << endl;
    cout << " เกรด P = " << P << " คน" << endl;
    cout << " เกรด G = " << G << " คน" << endl;
    return 0;
}
```

6.2 อาร์กิวเมนต์อาเรย์ (Array Arguments)

เราสามารถส่งอาเรย์ผ่านฟังก์ชันได้ โดยเขียนชื่อแต่ไม่ต้องมี subscript ในลิสต์ของอากิวเมนต์ของการเรียกฟังก์ชัน ซึ่งส่วนของ formal parameter list ของฟังก์ชันต้องมีการกำหนดพารามิเตอร์ให้สอดคล้องกับอากิวเมนต์ที่ส่งผ่านค่า โดยส่วนของพารามิเตอร์ต้องมีการกำหนด subscript เพื่อใช้อ้างถึงสมาชิกในอาเรย์

ตัวอย่างที่ 6.4 ฟังก์ชัน sameArray

ฟังก์ชันนี้เป็นตัวอย่างการส่งผ่านอาเรย์ 2 ชุดเพื่อทำการตรวจสอบว่าเหมือนกันหรือไม่ โดยข้อมูลที่เก็บในอาเรย์เป็นเลขทศนิยม จำนวน size ตัว ในกรณีที่เหมือนกันทุกค่าจะส่ง true กลับไปยังจุดเรียกใช้ แต่ถ้าไม่เท่ากันจะส่ง false กลับไปยังจุดเรียกใช้

```
// File : sameArray
// Compares two float arrays for equality by compareing corresponding elements
// Pre: a[I] and b[I] (0 <= I <= size-1) are assigned values.
// Post: Returns true if a[I] == b[I] for all I in range 0 through size-1 ;
// otherwise, return false

bool  sameArray ( float a[] ,                    // IN: float arrays to be compared
                  float b[] ,
                  int size)                      // IN: size of the arrays
{
    int I ;         // loop control variable and array subscript
    I = 0 ;
    while (( I < size-1) && (a[I] == b[I]))
                I++;
    return   (a[I]==b[I]);                       // default result
}
```

การเรียกใช้ฟังก์ชันนี้สามารถกระทำได้ดังนี้

```
if (sameArray(x,y,10))
    cout << "The arrays x and y are identical. " << endl ;
else
    cout << "The arrays x and y are different." << endl;
```

โดยกำหนดตัวแปร x และ y ดังนี้

```
    float  x[10];
    float  y[10];
```

ตัวอย่างที่ 6.5 ฟังก์ชัน addArray

ถ้าเราต้องการนำค่าของอาเรย์ 2 กลุ่มมาบวกกันเก็บในอาเรย์ตัวที่ 3 เราสามารถกำหนดฟังก์ชันและเรียกใช้ได้ดังนี้ addArray(size , x , y , z);

โดยข้อมูลที่เป็น input เก็บที่ตัวแปรอาเรย์ x และ y ส่วน ตัวแปรอาเรย์ z เก็บค่าผลลัพธ์จากการบวก สำหรับ size คือขนาดของข้อมูลที่เก็บในอาเรย์ ดังฟังก์ชันต่อไปนี้

```
// Stores the sum of a[i] and b[i] in c[i]
// Sums pairs of array elements with subscripts ranging from 0 to size-1
// Pre: a[i] and b[i] are defined (0 <= i <= size-1)
// Post: c[i] = a[i] + b[i]  (0 <= i <= size-1)
void addArray
        (int size,                          // IN: the size of the arrays
         const float a[],                   // IN: the first array
         const float b[],                   // IN: the second array
         float c[] )                        // OUT: result array
{       // Add corresponding elements of a and b and store in c.
        for (int i  = 0; i  <  size;  i++ )
                c[i] = a[i] + b[i];
}
```

เนื่องจากภาษา C++ มีการส่งผ่านอาเรย์เป็นแบบอ้างอิง(pass by reference) ดังนั้น อาเรย์ a , b , c จึงใช้ตำแหน่งจริงของอาเร ย์ x , y , z ซึ่งการทำงานของฟังก์ชันจะนำค่าของอาเรย์ x และ y มาบวกเก็บไว้ที่อาเรย์ z เพื่อป้องกันมิให้ค่าของอาเรย์ x และ y เปลี่ยนแปลงค่า เราใช้คำเฉพาะ (reserved word) const กำหนดไว้ข้างหน้า formal parameter ซึ่งสังเกตุจะเห็นว่าเราไม่กำหนดจำนวนสมาชิกในพารามิเตอร์ที่เป็นอาเรย์ เนื่องจากคอมไพเลอร์ไม่จำเป็นต้องจัดสรรเนื้อที่ในหน่วยความจำไม่ต้องมีการคัดลอกค่าข้อมูลต่างๆ ดังนั้นจึงไม่เตรียมเนื้อที่ขึ้นมาใหม่ สำหรับต้นแบบของการเรียกฟังก์ชันนี้สามารถกำหนดได้ดังนี้

```
void addArray(int , const float[] , const float[] , float[]);
```

เราสามารถสรุปถึงการส่งผ่านค่าที่เป็นอาเรย์ได้ดังนี้

- ภาษา C++ นั้นยินยอมให้มีการส่งผ่านโครงสร้างข้อมูลชนิดอาเรย์แบบอ้างอิง โดยไม่มีการสร้างเนื้อที่ขึ้นใหม่แต่อ้างอิงใช้เนื้อที่ที่มีการเรียกใช้ที่สอดคล้องกัน
- การกำหนดฟังก์ชัน หรือต้นแบบฟังก์ชัน ใช้ [] ว่าง เพื่อบอกคอมไพเลอร์ โดยไม่ต้องกำหนดถึงขนาดของอาเรย์
- คำเฉพาะ const ที่กำหนดหน้าพารามิเตอร์ แสดงให้ทราบว่าพารามิเตอร์นั้นไม่สามารถเปลี่ยนภายในฟังก์ชันได้ และถ้าพยายามแก้ไขข้อมูล คอมไพเลอร์จะแสดงข่าวสารของความผิดพลาดให้เราได้ทราบ
- สำหรับการเรียกใช้ (function call) เราเขียนชื่อ และอากิวเมนต์จริงโดยไม่ต้องใส่ [] หลังชื่อของตัวแปรอาเรย์

ตัวอย่างที่ 6.6 ฟังก์ชัน readScores

ในการแก้ปัญหาบางอย่างถ้าเราไม่ทราบถึงขนาดของข้อมูลที่เก็บในอาเรย์ เช่น การป้อนคะแนนสอบของนักศึกษาซึ่งในแต่ละห้องจะมีนักศึกษาจำนวนไม่เท่ากัน แต่การเขียนโปรแกรมต้องมีการกำหนดขนาดของอาเรย์ไว้ก่อนการประมวลผล เมื่อมีการป้อนคะแนนของนักศึกษาแต่ละคนจะมีการนับจำนวนเพิ่มขึ้นเรื่อยๆโดยเริ่มจาก 0 , 1 , ... จนถึงค่าสูงสุดที่กำหนดไว้

สำหรับตัวอย่างนี้เป็นการอ่านคะแนนของนักศึกษา โดยกำหนดให้ MAX_SIZE เป็นค่าคงที่ที่กำหนดไว้ โดยกำหนดไว้ก่อนดังนี้ const int MAX_SIZE = 500 ; เมื่อมีการนำมาอ้างอิงในฟังก์ชันจะมีการเขียนได้ดังนี้

```
// File: readScores
// Reads an array of exam scores for a lecture section of up to MAX_SIZE students.
// Pre: None
// Post: The data values are stores in array scores.
//       The number of values read is stored in sectionSize. (0 <= sectionSize <
MAX_SIZE)

void readScores
        (int scores [] ,                    // OUT: array to contain all scores read
         const int MAX_SIZE ,               // IN: max size of array scores
         int& sectionSize)                  // OUT: number of elements read
{
        // Local data    . . .
        const int SENTINEL = -1 ;           // sentinel value
        int tempScore;                      // temporary storage for each score
        // Read each array element until done.
        cout << "Enter next score after the prompt or enter"
             << SENTINEL  << " to stop. "  <<  endl;
        sectionSize  =  0;                  // initial class size
        cout << "Score: ";
        cin >> tempscore;
        while ((tempScore != SENTINEL) && (sectionSize < MAX_SIZE))
        {
                scores [sectionSize] = tempScore;          // save score just read
```

```
            sectionSize++;
            cout << "Score: ";
            cin >> tempScore;
        }       // end while
        // Sentinel was read or array is filled.
        if (tempScore != SENTINEL)
        {
            cout << "Array is filled!" << endl;
            cout << tempScore << " not stored " << endl;
        }
}
```

ตัวอย่างที่ 6.7 โปรแกรมตัดเกรด ปรับปรุงจากตัวอย่างที่ 6.3

ในการแก้ปัญหาโปรแกรมในตัวอย่างที่ 6.3 นั้นเราเขียนเป็นโปรแกรมเดี่ยวๆ ไม่มีการแบ่งปัญหาออกเป็นโมดูลย่อยๆ เมื่อเราทราบถึงวิธีการส่งผ่านค่าของข้อมูลชนิดอาเรย์แล้ว ทดลองมาเขียนเป็นลักษณะ Modular บ้าง ในที่นี้เราจะแบ่งโปรแกรมออกเป็นโมดูลต่างๆดังนี้

- INPUT1 ทำหน้าที่ในการรับจำนวนของนักศึกษาทั้งหมด เก็บในตัวแปร N
- INPUT2 ทำหน้าที่รับคะแนนของนักศึกษาจำนวน N คนเก็บในอาเรย์ชื่อ STU
- AVERAGE ทำหน้าที่หาคะแนนเฉลี่ยของคะแนนสอบของนักศึกษาจำนวน N คน
- MAXIMUM ทำหน้าที่หาค่าคะแนนที่มากที่สุดของกลุ่ม
- COUNT ทำหน้าที่นับจำนวนของนักศึกษาที่ได้เกรด F , P และ G
- PRINT ทำหน้าที่พิมพ์ผลลัพธ์จากการทำงานออกทางจอภาพ

```
#include <iostream>
using namespace std;
void INPUT1(int &);
void INPUT2(int , int [] );
```

```
int  AVERAGE(int , const int [] );
int MAXIMUM (int , const int [] );
void COUNT(int , const int [] , int & , int & ,int & );
void PRINT(int , int  , float , int , int , int );
int main ()
{
 int  STU[1000] ;
 int N ;
 int  MAX ;
 float  AVG ;
 int F=0 ; P=0 ; G=0 ;
 INPUT1(N);
 INPUT2(N , STU);
 MAX = MAXIMUM(N , STU);
 AVG = AVERAGE(N , STU);
  COUNT(N , STU , F , P , G );
  PRINT(N , MAX , AVG , F , P ,G );
  return 0;
}
void INPUT1(int & N)
{
    cout << "จำนวนของนักศึกษาทั้งหมด : " ;  cin >> N ;
}
// ฟังก์ชันรับคะแนนสอบของนักศึกษาจำนวน n คน
void INPUT2(int  n , int  SCORE[] )
{
  for ( int  I=0 ; I< n ;I++)
```

```
    {
        cout << " คะแนนของนักศึกษาคนที่ " << I+1 << "=" ;
        cin >> SCORE[I] ;
    }
}
// ฟังก์ชันหาค่าเฉลี่ยของคะแนนสอบ
int AVERAGE(int n , const int SCORE[] )
{
    float SUM , AVG ;
    SUM = 0.0 ;
    for(I=0 ; I<n ; I ++)
        SUM = SUM + SCORE[I];
    AVG = SUM / N ;
    return AVG ;
}
// ฟังก์ชันหาค่ามากที่สุดของคะแนนสอบ
int MAXIMUM (int n , const int SCORE[] )
{
    int MAX , I ;
    MAX = SCORE[0] ;
    for (I=1 ; I<n ; I++)
    {
        if (MAX < SCORE[I])
            MAX = SCORE[I] ;
    }
    return MAX;
}
```

```
//หาจำนวนของเกรดต่างๆ
void COUNT(int  n , const int SCORE [] , int  & F  , int & P  ,int & G)
{
    for (int  I=0 ; I<N ; I++)
    {
       if (SCORE[I] <60)
                F++ ;
       else if  (SCORE[I] < 80 )
                P++;
       else
                G++ ;
    }
}
// พิมพ์ผลลัพธ์
void PRINT(int N , int  MAX , float  AVG , Int  F , int  P , int  G)
{
    cout << "จำนวนของนักศึกษาที่ลงทะเบียนทั้งหมด = " << N << "คน" << endl ;
    cout << "คะแนนสูงสุดคือ " << MAX << endl;
    cout << " คะแนนเฉลี่ยคือ " << AVG << endl;
    cout << " เกรด F = " << F << " คน" << endl;
    cout << " เกรด P = " << P << " คน" << endl;
    cout << " เกรด G = " << G << " คน" << endl;
}
```

6.3 การค้นหาและการเรียงลำดับข้อมูล

การค้นหาและการเรียงลำดับข้อมูลเป็นกระบวนการพื้นฐานที่กระทำกับโครงสร้างข้อมูลชนิดอาเรย์ ซึ่งในการปฏิบัติงานในชีวิตประจำวันเราจะไม่พ้นการค้นหาข้อมูลที่ต้องการด้วยกุญแจต่างๆ เมื่อไปห้างสรรพสินค้าเราจะเห็นพนักงานขายสินค้าโดยการป้อนรหัสสินค้า เพื่อค้นหา

ราคาสินค้า โดยนำมาคิดราคาให้กับลูกค้า ตัวเราเองก็มีการถอนเงินผ่านตู้เอทีเอ็มโดยการป้อนรหัสผ่าน เพื่อค้นหาบัญชีของเรา เรามาทำความเข้าใจกับกระบวนการค้นหาข้อมูลกันดีกว่า

ตัวอย่างที่ 6.8 ฟังก์ชันในการหาค่าน้อยที่สุดในอาเรย์

ตัวอย่างนี้เป็นการสร้างฟังก์ชันในการหาค่าที่น้อยที่สุดในตัวแปรอาเรย์ x โดยกำหนดให้ตัวแปร startIndex เป็นตัวแปรที่เก็บตำแหน่งเริ่มต้นของข้อมูลที่ต้องการค้นหา และ endIndex เป็นตัวแปรที่เก็บตำแหน่งสุดท้ายของข้อมูลที่ต้องการค้นหา ผลจากการทำงานจะได้ตำแหน่งของข้อมูลที่น้อยที่สุดในช่วงที่กำหนด

```
int findIndexOfMin
        (const float x[],          // IN: array of elements
         int startIndex,           // IN: subscript of first element
         int endIndex)             // IN: subscript of last element
{
        // Local data  . . .
        int minIndex;                       // index of the smallest element found
        int i;                              // index of the current element
        // Validate subarray bounds
        if ((startIndex < 0)  ||  (startIndex > endIndex))
        {
                cerr << "Error in subarray bounds" << endl;
                return – 1;                 // return error indicator
        }
        minIndex = startIndex;
        for (i = startIndex + 1; i <= endIndex; i++)
                if (x[i] < x[minIndex])
                        minIndex = i;
```

```
    return minIndex;                        // return result
}   // end findIndexOfMin
```

การทำงานของฟังก์ชันนี้นั้นจะทำการหาตำแหน่งของข้อมูลที่เป็นเลขทศนิยมที่น้อยที่สุด โดยตรวจสอบค่าที่ส่งมา ถ้า ค่าของ startIndex น้อยกว่า 0 หรือ มากกว่า endIndex จะไม่กระทำงานโดยแสดงข้อผิดพลาดออกมาทางจอภาพดังนี้ " Error in subarray bounds " และส่งค่า –1 กลับไปยังจุดเรียกใช้ แต่ถ้าเป็นข้อมูลที่ถูกต้องจะทำการค้นหาตำแหน่งของข้อมูล โดยสมมุติให้ตำแหน่งของข้อมูลตัวแรก เป็นตำแหน่งที่มีค่าที่เก็บน้อยที่สุดก่อน ต่อจากนั้นใช้การเปรียบเทียบกับข้อมูลในตำแหน่งที่เหลือ โดยใช้คำสั่งการทำงานวนรอบ for ถ้า ข้อมูลที่เปรียบเทียบเกิดน้อยกว่า ต้องเปลี่ยนให้ตำแหน่งข้อมูลที่เก็บค่าน้อยที่สุดเป็นตำแหน่งใหม่นั้น โดยนำตำแหน่งใหม่แทนที่ตำแหน่งเก่า กระทำการวนรอบเปรียบเทียบจนครบทุกๆตัว เราจะได้ตำแหน่งของข้อมูลตัวที่น้อยที่สุด และส่งกลับไปยังจุดเรียกใช้ ในที่นี้เก็บในตัวแปร minIndex นั่นเอง

ตัวอย่างที่ 6.9 การค้นหาแบบลำดับ(linear search)

ตัวอย่างนี้เป็นการสร้างฟังก์ชันในการค้นหาข้อมูล โดยส่งผ่านกลุ่มของข้อมูลทั้งหมดโดยอ้างอิงแบบคงที่ผ่านตัวแปรอาเรย์ชื่อ item นอกจากนี้มีการส่งกุญแจหรือข้อมูลที่ต้องการค้นหาโดยเก็บในตัวแปร target และส่งขนาดของอาเรย์เก็บในตัวแปร size การค้นหาของฟังก์ชันนี้จะส่งผ่านตำแหน่งของข้อมูลที่ค้นหากลับไปยังจุดเรียกใช้ ในกรณีที่ไม่พบจะส่ง –1 กลับ

```
int linSearch
        (const int items[],             // IN: the array being searched
         int target,                    // IN: the target being sought
         int size)                      // IN: the size of the array
{
        for (int next = 0, i < size, i++)
                if (items[next]  = = target)
```

```
            return next;                    // found, return subscript
        // Assertion: All elements were tested without success.
        return – 1;
}       // end linSearch
```

ตัวอย่างที่ 6.10 Selection Sort

ตัวอย่างนี้เป็นการสร้างฟังก์ชันในการเรียงลำดับข้อมูลในอาเรย์ โดยใช้หลักการในการค้นหาข้อมูลมาไว้ในตำแหน่งที่เหมาะสม

สมมุติว่าข้อมูลก่อนเรียงลำดับมีดังนี้

[0]	[1]	[2]	[3]
74	45	83	16

ถ้าเราต้องการเรียงลำดับข้อมูลใหม่จากน้อยไปมาก จะเรียงลำดับเลยทีเดียวไม่ได้ต้องกระทำเป็นขั้นตอนโดยต้องหาให้ได้ว่าข้อมูลตัวที่น้อยที่สุดอยู่ในตำแหน่งไหน แล้วนำมาไว้เป็นลำดับแรกในที่นี้ จะเห็นได้ว่า ข้อมูล 16 ในตำแหน่งที่ 3 น้อยที่สุด ต้องนำมาไว้ในตำแหน่ง แรก แต่ไม่ลืมที่จะนำข้อมูล 74 ไปไว้ในตำแหน่งที่เหมาะสม ซึ่งเราก็ไม่รู้ว่าตำแหน่งไหน ทางที่ดีที่สุดคือ สลับข้อมูลกันระหว่างตำแหน่งที่ 0 กับ 3

[0]	[1]	[2]	[3]
16	45	83	74

ซึ่งการกระทำในขั้นตอนแรกนั้นเราจะได้ข้อมูลตัวที่น้อยที่สุดอยู่ในตำแหน่งที่ 0 จะเห็นว่าข้อมูลที่เหลือยังไม่ได้สลับ ทำอย่างไรดี? ก็กระทำแบบเดียวกันโดยสนใจเฉพาะข้อมูลตำแหน่งที่ 1 ถึง ตำแหน่งที่ 3 เท่านั้น เมื่อพิจารณาจะเห็นว่า ข้อมูลตัวที่น้อยที่สุดคือ 45 อยู่ในตำแหน่งที่ 1 ซึ่งเป็นตำแหน่งที่เหมาะสมอยู่แล้ว

สำหรับขั้นตอนต่อไปเราจะพิจารณาข้อมูลที่เหลือคือตำแหน่งที่ 2 และตำแหน่งที่ 3 ซึ่งข้อมูลตัวที่น้อยอยู่ในตำแหน่งที่ 3 ต้องสลับค่าระหว่างข้อมูลในตำแหน่งที่ 2 และตำแหน่งที่ 3

[0]	[1]	[2]	[3]
16	45	74	83

สังเกตุได้ว่าในขั้นตอนแรกข้อมูลที่น้อยที่สุดอยู่ในตำแหน่งที่ 0 ในขั้นตอนต่อไปข้อมูลตัวที่น้อยเป็นลำดับถัดมาอยู่ในตำแหน่งที่ 1 ขั้นตอนต่อไปข้อมูลตัวที่น้อยเป็นลำดับถัดมาอีกอยู่ในตำแหน่งที่ 2 ดังนั้นสรุปได้ว่าถ้ามีข้อมูล 4 ตัวจะกระทำ 3 รอบ เพราะตัวสุดท้ายไม่ต้องไปเปรียบเทียบกับข้อมูลตัวใดๆ เราสามารถสรุปอัลกอริทึมในการเรียงลำดับโดยวิธีการนี้ได้ดังนี้

- กำหนดให้ i เป็นตัวแปรที่ระบุถึงตำแหน่งของข้อมูลที่มีค่าน้อยที่สุดในการทำงานแต่ละรอบ ในที่นี้ถ้า i เท่ากับ 0 ค่าที่น้อยที่สุดในการค้นหาจะอยู่ในตำแหน่งที่ 0 ค่าของ i จะเพิ่มขึ้นรอบละ 1 จนกระทั่งมีค่าเท่ากับ n-1 กำหนดให้ n คือจำนวนของข้อมูลทั้งหมดในอาเรย์
- การทำงานแต่ละรอบจะทำการค้นหาว่าข้อมูลที่น้อยที่สุดอยู่ในตำแหน่งใด ระบุถึงขอบเขตของข้อมูลที่ต้องการค้นหา โดยเรียกใช้ฟังก์ชัน findIndexOfMin ในตัวอย่างที่ 6.8
- เมื่อได้ตำแหน่งของข้อมูลตัวที่น้อยที่สุดแล้วต้องนำไปไว้ในตำแหน่งที่ถูกต้องโดยตรวจสอบกับค่าของ i ถ้าเหมือนกันแสดงว่าอยู่ในตำแหน่งที่ถูกต้องแล้ว แต่ถ้าไม่เหมือนกันต้องทำการสลับตำแหน่งในที่นี้เราสามารถสร้างฟังก์ชันในการสลับค่า ชื่อว่า exchange ได้ดังนี้

```
void  exchange(int &  a  , int & b )
{
        int temp;
        temp   = a ;
        a      = b ;
        b      = temp ;
}
```

- กระทำซ้ำโดยกระทำจนครบทุกรอบ จะได้ข้อมูลเรียงลำดับจากน้อยไปมากตามต้องการ

```
void selSort(int items[], int n)
{
        // Local data    . . .
        int minSub;                    // subscript of each smallest item located by
                                       // findIndexOfMin
        for (int i = 0; i < n – 1; i++)
```

```
    {
        minSub = findIndexOfMin(items, i , n - 1);
        exchange (items[minSub], items[i]);
    }
}
```

กรณีศึกษา : การประมวลผลอาร์เรย์ 1 มิติ

ปัญหา ต้องการรับคะแนนสอบ วิชา CT 212 ของนักศึกษา จำนวน n คน เก็บในตัวแปรอาร์เรย์ ชื่อ stu สิ่งที่ต้องการ

1.1 หาผลรวมของคะแนนสอบทั้งหมด

1.2 หาค่าเฉลี่ยของคะแนนสอบ

1.3 หาคะแนนสอบที่มีค่ามากที่สุด

1.4 เรียงลำดับคะแนนสอบจากมากไปหาน้อย

1.5 การสืบค้นข้อมูลที่ต้องการ

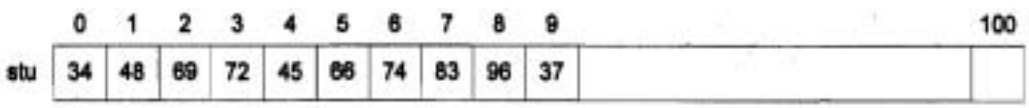
เมื่อมีการ Run โปรแกรม จะให้ผู้ใช้ป้อนจำนวนนักศึกษาและป้อนคะแนนสอบของนักศึกษา เรียงลำดับตั้งแต่คนแรก จนถึงคนสุดท้ายดังนี้

ป้อนจำนวนนักศึกษา = 10 คน

ป้อนคะแนน

34 48 69 72 45 66 74 83 96 37

การทำงานของโปรแกรมจะนำไปเก็บในตัวแปร stu โดยเริ่มจากตัวชี้ที่ 0 เรียงจนถึงตัวชี้ที่ 9 ดังรูป

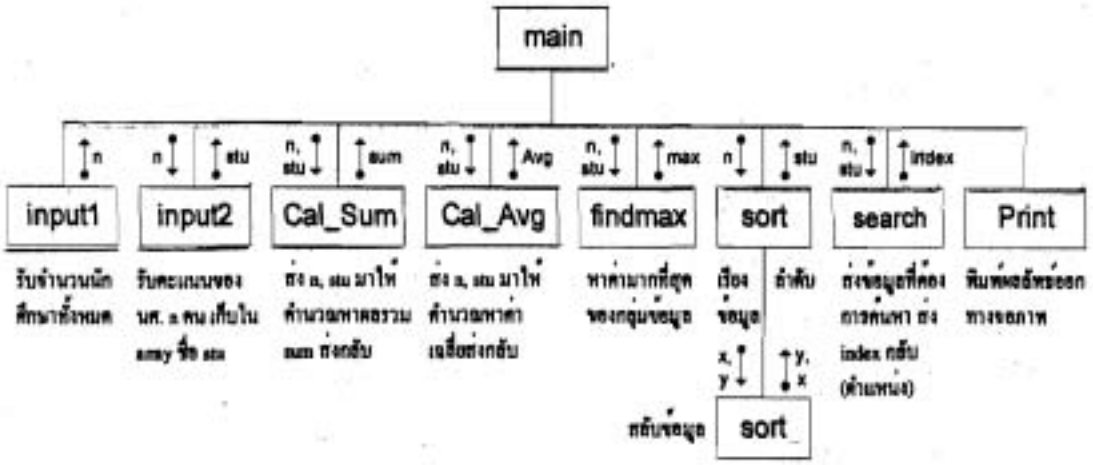


```
#include <iostream>
using namespace std;
int main ()
{
    int n;              //จำนวนนักศึกษาทั้งหมด
    int stu[100];       //เก็บคะแนนของนักศึกษาเป็นอาร์เรย์ 1 มิติ ขนาดสูงสุด 100 ช่อง
    int sum;            //หาผลรวมคะแนนทั้งหมด
    double avg;         //ค่าเฉลี่ยของคะแนน
    bool found;
    int max_score;      //คะแนนที่มากที่สุด
    int search_data;    //ข้อมูลที่ต้องการสืบค้น
    int index;          //เก็บตัวชี้ที่สืบค้นพบ
    int i;
    int max1,temp;
    cout<<"Total student = ";
    cin>>n;
    for(i=0;i<n;i++)
        cin>>stu[i];
    sum=0;
    for(i=0;i<n;i++)
        sum=sum+stu[i];
    avg=sum/n;
// หาคะแนนที่มากที่สุด
max=stu[0];
    for(i=1;i<n;i++)
    {
        if(max<stu[i])
```

```
                max=stu[i];
        }
// เรียงลำดับข้อมูล
for(i=0;i<n;i++)
        {
                max1=i;
                for(int j=0;j<n;j++)
                {
                        if(stu[max]<stu[j])
                                max=j;
                }
                if(max1!=i)
                {
                        temp=stu[max1];
                        stu[max1]=stu[i];
                        stu[i]=temp;
                }
        }
// สืบค้นข้อมูล
        i=0;
        cin>>search_data;
        found=false;
        while(!found&&i<n)
        {
                if(search_data==stu[i])
                        found=true;
                else i++;
```

```
    }
    if(found)
        index=i;
    else
    cout<<"not found"<<endl;
    return 0;
}
```

เราสามารถเขียนโปรแกรมนี้ทำในรูปแบบฟังก์ชั่น ได้ดังนี้




```
#include <iostream>
using namespace std;
void Input1(int&);
void Input2(int,int[]);
int CalSum(int,const int[]);
double CalAvg(int,int);
int Findmax(int,const int[]);
void Sort(int,int[]);
```

```
void Swap(int&,int&);
int Search(int,int,const int[]);
void Print(int,double,int);
int main()
{
    int n ;
    int stu[100];
    int sum;
    double Avg;
    int max;
    int index;
    int key;
    Input1(n);
    Input2(n,stu);
    sum=CalSum(n,stu);
    Avg=CalAvg(n,sum);
    max=Findmax(n,stu);
    Sort(n,stu);
    cout<<"Please put key ";
    cin>>key;
    index=Search(key,n,stu);
    if(index!=-1)
        cout<<"Not found"<<endl;
    Print(sum,Avg,max);

    return 0;
}
```

```
void Input1(int& x)
{
    cin>>x;
}
void Input2(int n,int A[])
{
    int i;
    for(i=0;i<n;i++)
        cin>>A[i];
}
int CalSum(int n,const int A[])
{
    int i;
    int s=0;
    for(i=0;i<n;i++)
        s=s+A[i];
    return s;
}
double CalAvg(int n,int sum)
{
    return sum/n;
}
int Findmax(int n,const int A[])
{
    int i;
    int m;
    m=A[0];
```

```
    for(i=0;i<n;i++)
        if(m<A[i])
            m=A[i];
    return m;
}
void Sort(int n,intA[])
{
    int i,j;
    int m;
    for(i=0;i<n;i++)
    {
        m=i;
        for(j=i+1;j<n;j++)
        {
            if(A[m]<A[j])
                m=j;
        }
        if(i!=m)
            Swap(A[i],A[m]);
    }
}
void Swap(int& A,int& B)
{
    int temp;
    temp=A;
    A=B;
    B=temp;
```

```
}
int Search(int key,int n,const int A[])
{
    int i;
    for(i=0;i<n;i++)
    {
        if(key=A[i])
            return i;
    }
    return -1;
}
void Print(int sum,double Avg,int max)
{
    cout<<"Sum = "<<sum<<endl;
    cout<<"Average = "<<Avg<<endl;
    cout<<"Max = "<<max<<endl;
}
```

การส่งผ่านข้อมูลชนิดอาร์เรย์ 1 มิติ โดยทั่วไปจะเป็นการส่งผ่านแบบ PASS BY REFERENCE ในกรณีที่ต้องการส่งผ่านแบบ PASS BY VALUE ต้องกำหนดคำเฉพาะ const ก่อนหน้า Formal parameter นั้น

จากตัวอย่างนี้มีฟังก์ชั่นที่ส่งผ่านแบบ PASS BY REFERENCE ได้แก่

```
void Input1(int&);
void Input2(int,int[]);
void Sort(int,int[]);
void Swap(int&,int&);
```

จากตัวอย่างนี้มีฟังก์ชั่นที่ส่งผ่านแบบ PASS BY VALUE ได้แก่

```
int CalSum(int,const int[]);
double CalAvg(int,int);
int Findmax(int,const int[]);
int Search(int,int,const int[]);
void Print(int sum,double Avg,int max)
```

6.4 อาเรย์สองมิติ(Two Dimension Arrays)

เป็นการกำหนดตำแหน่งของข้อมูลในลักษณะ 2 มิติ กล่าวคือมีตัวชี้ 2 ตัว มีรูปแบบดังนี้

```
datatype  arrayName[intExp1][intExp2] ;
```

โดย intExp1 และ intExp2 ต้องเป็นค่าคงที่ที่มีค่าเป็นเลขจำนวนเต็มบวกเท่านั้น ซึ่งเป็นตัวชี้ของข้อมูล โดยเปรียบ intExp1 เป็นแถว และ intExp2 เป็นคอลัมภ์ เช่น float matrix[2][3]; เครื่องคอมพิวเตอร์จะจัดสรรเนื้อที่ว่างสำหรับเก็บค่าที่เป็นทศนิยม 2x3 = 6 เนื้อที่ดังนี้

matrix

	[0]	[1]	[2]
[0]			
[1]			

ในที่นี้ตัวแปรอาเรย์ matrix ประกอบด้วย 2 แถว คือ แถว0 และแถว1 และ 3 คอลัมภ์ คือคอลัมภ์ 0 , คอลัมภ์ 1 และ คอลัมภ์ 2 เราสามารถนำข้อมูลไปเก็บในตัวแปรอาเรย์นี้ได้ ต้องมีการอ้างถึงตำแหน่งของแถวและคอลัมภ์ โดยมีรูปแบบดังนี้

```
arrayName[indexExp1][indexExp2]
```

เช่น คำสั่ง

```
matrix[1][1] = 2.5
```

มีผลให้นำข้อมูล 2.5 ไปเก็บในตัวแปร matrix ในแถวที่ 1 และคอลัมภ์ที่ 1 ดังนี้

matrix

	[0]	[1]	[2]
[0]			
[1]		2.5	

เราสามารถกำหนดค่าเริ่มต้นให้แก่ตัวแปรอาเรย์ สองมิติได้ ดังนี้

```
const int NUM_ROWS = 2;
const int NUM_COLS = 3;
float matrix [NUM_ROWS] [NUM_COLS] = {{5.0, 4.5, 3.0}, {-6.0, 2.5, 1.0}};
```

มีผลให้ค่าต่างๆถูกนำไปเก็บในตัวแปร matrix ตามลำดับดังนี้

matrix

	[0]	[1]	[2]
[0]	5.0	4.5	3.0
[1]	-6.0	2.5	1.0

ตัวอย่างที่ 6.11 การหาผลรวมของข้อมูลใน MATRIX

ตัวอย่างนี้เป็นการสร้างฟังก์ชันชื่อ sumMatrix ซึ่งมีการส่งผ่านกลุ่มข้อมูลที่มีโครงสร้างเป็น Array 2 มิติ ขนาด rows * NUM_COLS

```
// File: sumMatrix.cpp
// Calculates the sum of the elements in the first rows
//        of an array of floating point values
//        with NUM_COLS  (a constant)  columns.
// Pre: The type int constant NUM_COLS is defined (NUM_COLS > 0)
        and the array element values are defined and rows > 0.
// Post:  sum is the sum of all array element values.
// Returns:    Sum of all values stored in the array.
```

```
float sumMatrix
    (float table [ ] [NUM_COLS] ,        // IN:  array to be summed
     int rows)                           // IN:  number of  rows in array
                                         //      (rows > 0)
{
      float sum = 0.0; // sum of all element values
                       //    - initially 0.0

      // Add each array element value to sum.
      for  (int r = 0;  r < rows;  r++)
        for  (int c = 0;  c < NUM_COLS;  c++)
           sum += table [r] [c];

      return sum;
}
```

กรณีศึกษา :

การประมวลผลเมตริกซ์

ปัญหา ต้องการรับข้อมูลเป็น Matrix ขนาด 3×4 เก็บในตัวแปรชื่อ A
โดยผู้ใช้ต้องป้อนข้อมูลทั้งหมด 12 จำนวนทางแป้นพิมพ์เพื่อนำไปเก็บในหน่วยความจำหลัก

8 4 6 7 8 9 1 0 1 0 1↵

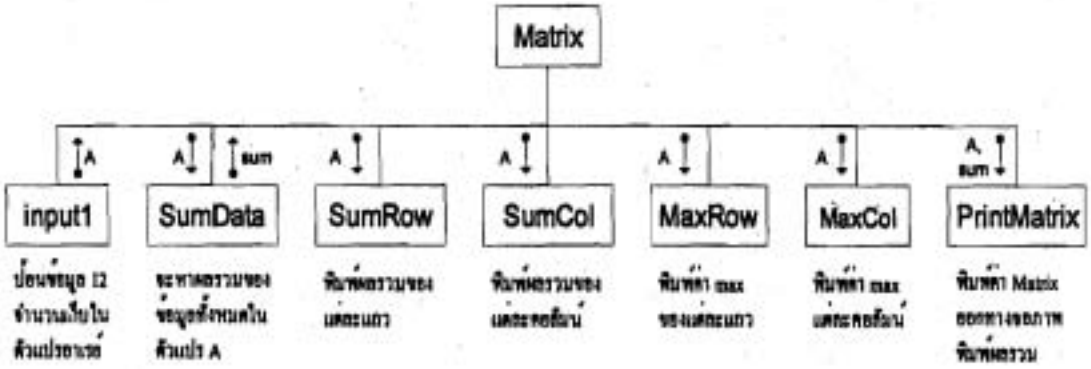
สิ่งที่ต้องการ

1. ผลรวมทั้งหมดของข้อมูล
2. ผลรวมของแถว และคอลัมน์
3. ค่ามากที่สุดของแถว และคอลัมน์

```
#include <iostream>
using namespace std;

int main()
{
    int A[3][4];
    int row,col;
    int sum,sumrow,sumcol,max;

    for(row=0;row<3;row++)
    {
        for(col=0;col<4;col++)
            cin>>A[row][col];
    }

    sum=0;
    for(row=0;row<3;row++)
        for(col=0;col<4;col++)
            sum=sum+A[row][col];

    for(row=0;row<3;row++)
    {
        sumrow=0;
        for(col=0;col<4;col++)
            sumrow=sumrow+A[row][col];
        cout<<"Sum of row "<<row+1<<"= "<<sumrow<<endl;
    }
```

```
    for(col=0;col<4;col++)
    {
        sumcol=0;
        for(row=0;row<3;row++)
            sumcol=sumcol+A[row][col];
        cout<<"Sum of column "<<col+1<<"= "<<sumcol<<endl;
    }
    for(row=0;row<3;row++)
    {
        max=A[row][0];
        for(col=1;col<4;col++)
            if(max<A[row][col])
                max=A[row][col];
       cout<<"Max of row "<<row+1<<"= "<<max<<endl;
    }
    for(col=0;col<4;col++)
    {
        max=A[0][col];
        for(row=1;row<3;row++)
            if(max<A[row][col])
                max=A[row][col];
        cout<<"Max of column "<<col+1<<"= "<<max<<endl;
    }
    return 0;
}
```

เขียนในรูปแบบ Module Function




```
#include <iostream>
using namespace std;
void Input1(int[][4];
int SumData(const int[][4]);
void SumRow(const int[][4]);
void SumCol(const int[][4]);
void MaxRow(const int[][4]);
void MaxCol(const int[][4]);
void PrintMatrix(const int[][4],int);
int main()
{
    int A[3][4];
    int sum;
    Input(A);
    sum=SumData(A);
    PrintMatrix(A,sum);
    SumRow(A);
    SumCol(A);
    MaxRow(A);
    MaxCol(A);
```

```
    return 0;
}
void Input(int A[][4])
{
    int row,col;
    for(row=0;row<3;row++)
        for(col=0;col<4;col++)
            cin>>A[row][col];
}
int SumData(const int x[][4])
{
    int row,col;
    int sum=0;
    for(row=0;row<3;row++)
        for(col=0;col<4;col++)
            sum=sum+x[row][col];
    return sum;
}
void SumRow(const int x[][4])
{
    int row,col , sr;
    for(row=0,row<3;row++) {
        sr=0;
        for (col=0;col<4;col++)
            sr=sr+x[row][col];
        cout<<"Sum of row "<<row+1<<"= "<<sr<<endl; }
}
```

```
void SumCol(const int x[][4])
{
    int row,col;
    int sc=0;
    for(col=0;col<4;col++)
        for(row=0;row<3;row++)
            sc=sc+x[row][col];
    cout<<"Sum of column "<<col+1<<"= "<<sc<<endl;
}
void MaxRow(const int x[][4])
{
    int row,col;
    int max=0;
    for(row=0;row<3;rpw++)
    {
        max=A[row][0];
        for(col=1;col<4;col++)
            if(max<A[row][col])
                max=A[row][col];
        cout<<"Max of row "<<row+1<<"= "<<max<<endl;
    }
}
void MaxCol(const int x[][4])
{
    int row,col;
    int max=0;
    for(col=0;col<4;col++)
```

```
    {
        max=A[0][col];
        for(row=1;row<3;row++)
            if(max<A[row][col])
                max=A[row][col];
        cout<<"Max of column "<<col+1<<"= "<<max<<endl;
    }
}
void PrintMatrix(const int x[][4],int sum)
{
    int row,col;
    for(row=0;row<3;row++)
    {
        for(col=0;col<4;col++)
            cout<<x[row][col]<<" ";
        cout<<endl;
    }
    cout<<"Total sum = "<<sum<<endl;
}
```

กรณี อาเรย์ 2 มิติ จะใช้วิธีการส่งผ่านอาเรย์2มิติ โดยใช้คำสั่งต่อไปนี้ไม่ได้

```
void Input(int A[ ][ ] )
```

ต้องมีการระบุขนาดเข้าไปด้วย เช่น [3][4] หรือ [ ][4] จึงสามารถอ้างอิงได้ในฟังก์ชัน

```
void Input(int A[ 3][4] )
void Input(int A[ ][4] )
```

กรณีของ Array โดยทั่วไปจะส่งผ่านค่าแบบ PASS BY REFERENCE คือการปฏิบัติงานในฟังก์ชั่น จะส่งผลกับข้อมูลที่เก็บใน Array แต่ถ้าโปรแกรมเมอร์ไม่ต้องการให้ข้อมูลในอาเรย์ เปลี่ยน

parameter ที่เป็นโครงสร้างชนิด Array จากตัวอย่างนี้ ฟังก์ชั่นที่มีการส่งผ่าน แบบ PASS BY VALUE ได้แก่

```
int SumData(const int[][4]);
void SumRow(const int[][4]);
void SumCol(const int[][4]);
void MaxRow(const int[][4]);
void MaxCol(const int[][4]);
void PrintMatrix(const int[][4],int);
```

ส่วน PASS BY REFERENCE ได้แก่ void Input1(int[][4]; ภาษา C++ นั้นเราสามารถกำหนด Array ได้หลายมิติ ขึ้นอยู่กับการแก้ปัญหาในงานประยุกต์ชนิดต่างๆ ตามความเหมาะสม ในที่นี้จะยกตัวอย่างการใช้ Array 3 มิติ ในการแก้ปัญหา เราสามารถกำหนด Array ที่ชื่อว่า sales ดังนี้

```
const int PEOPLE = 10;
const int YEARS =5;
double sales[PEOPLE] [YEARS] [12];
```

จากตัวอย่างที่กำหนดข้างต้น ตัวแปร sales จะถูกจัดสรรข้อมูล 600 ช่อง ต่อเนื่องกันไป ซึ่งได้จากการนำ PEOPLE*YEARS*12 ก็คือ 10*5*12 มีค่า = 600 นั่นเอง โดยsales [0] [2] [11] จะแทนการขายของพนักงานขายคนที่ 1 ซึ่งมี subscript เป็น 0 ในช่วงของเดือนธันวาคม ซึ่งมี subscript เป็น 11 ของปีที่ 3 ซึ่งมี subscript เป็น 2

ตัวอย่างที่ 6.12 การหายอดขายรวมของพนักงาน 10 คน

ตัวอย่างนี้เป็นคำสั่งในการหาและแสดงยอดรวมของการขายของพนักงาน 10 คน

```
// Find and display the total dollar amount of sales by person
for (int person = 0; person < PEOPLE; person++)
{
 totalSales = 0.0;
 // Find the total sales for the current person.
```

```
    for (int year = 0; year < YEARS; year++)
        for (int month = 0; month < 12; month++)
            totalSales += sales [person] [year] [month];
    cout << "Total sales amount for salesperson"
         << person << " is " << totalSales << endl;
}
```

6.5 ข้อมูลชนิด Structs

โครงสร้างข้อมูลชนิด Array จะมีประโยชน์สำหรับการจัดการข้อมูลที่มีชนิดเดียวกันเช่น เก็บกลุ่มรองคะแนนของนักศึกษา 10 คน โดยข้อมูลใน Array ทั้งหมดเป็น Integer แต่ถ้าเราต้องการเก็บข้อมูลที่มีชนิดข้อมูลที่แตกต่างกัน เช่นข้อมูลบุคคลประกอบด้วย รหัส,ชื่อ,เพศ,อายุ,สถานะภาพ ซึ่งมีข้อมูลต่างชนิดกัน เราสามารถกำหนดข้อมูลชนิด Structs เพื่อแก้ปัญหานี้ได้ โดยองค์ประกอบต่าง ๆ นี้ ก็คือสมาชิก (member) ของ Structs นั่นเอง

struct Type Declaration	

รูปแบบ:	struct *struct-type*
	{
	Type$_1$ id-list$_1$;
	Type$_2$ id-list$_2$;
	
	type $_n$ id-list $_n$;
	};

ตัวอย่าง:

```
sturct complex
{
        float realPart ;
        float imaginaryPart;
 };
```

ความหมาย : เป็นการกำหนดโครงสร้างชนิดข้อมูลชนิด Struct โดย Sturct-type คือชื่อของโครงสร้าง ที่ผู้ใช้กำหนดขึ้น id-list คือ รายการข้อมูลในโครงสร้าง

ตัวอย่างที่ 6.13 การประกาศตัวแปรชนิด Struct

สมมุติผู้จัดการต้องการจัดเก็บข้อมูลพนักงานซึ่งมีโครงสร้างข้อมูลดังนี้

ID: 1234
NAME:Noel Goddard
Gender:Female
Number of dependents:0
Hourly rate:12.00
Total wages:480.00

เราสามารถกำหนดชนิดข้อมูลใหม่เป็นโครงสร้างชนิด Struct ชื่อ employee ประกอบด้วยสมาชิก 6 รายการข้อมูล ดังนี้

```
Struct employee
{
        int id;
        string name;
        char gender;
        int numDepend;
        float rate;
```

```
        float totWages;
};
```

โดยโครงสร้างนี้เป็นการกำหนดให้ Complier นั้นได้ทราบ แต่ยังไม่มีการจัดสรรเนื้อที่แต่อย่างใด เราสามารถจัดสรรเนื้อที่โดยกำหนดตัวแปรใหม่เพื่อเก็บข้อมูลในโครงสร้างนี้ได้ ดังนี้

```
employee  organist, janitor;
```

ตัวแปร organist และ janitor เป็นตัวแปรชนิด employee ซึ่งจะถูกจัดสรรเนื้อที่ให้มีขนาดเท่ากันสามารถเก็บรายการข้อมูลได้ 6 รายการข้อมูล เราสามารถเข้าถึงสมาชิกของโครงสร้างโดยอ้างสมาชิกของตัวแปรโดยใช้เครื่อง (.) ดังนี้

```
organist.id = 1234;
organist.name = "Noel Goddard";
organist.gender = 'F';
organist.numDepend = 0;
organist.rate = 12.00;
organist.totWages = 480.0;
```

เราสามารถนำข้อมูลมาประมวลผลโดยใช้คำสั่ง กำหนดค่า,คำสั่งทางเลือก หรืออื่นๆ ตามความเหมาะสม ได้ดังนี้

```
organist.totWages += (organist.rate*40.0);
```

หรือ

```
cout << "The organist is ";
switch (organist.gender)
{
        case 'F' : case 'f':
           cout <<  "Ms.  ";
            break;
        case  'M' : case  'm':
           cout << "Mr.  ";
           break;
        default:
```

```
            cout<< organist.gender
                <<" is bad character for gender!"<<endl;
        }
        cout<< organist.name << endl;
```

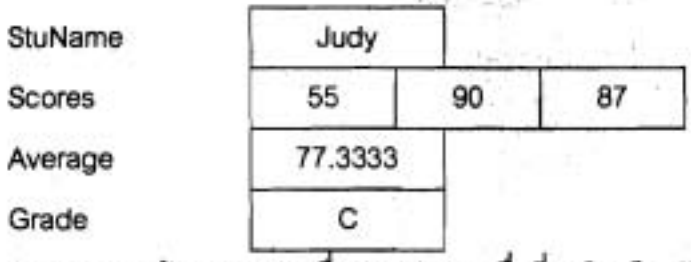
6.6 Struct as Operands and Arguments

โปรแกรมเมอร์บางครั้งมีความจำเป็นที่จะต้องประมวลผลข้อมูลชนิด Sturct โดยมีการส่งผ่านข้อมูลชนิด Struct ไปยัง function ชนิดต่างๆ โดยต้องจัดเตรียม actual Arguments ให้มีชนิดเดียวกันกับ formal parameter

สมมุติต้องการเก็บข้อมูลของนักศึกษาประกอบด้วยคะแนนสอบ 3 ครั้ง, คะแนนเฉลี่ย และเกรด เราสามารถกำหนดโครงสร้าง Sturct ที่ชื่อ examStats ได้ดังนี้

```
struct examStats
{
        string stuName;
        int scores[3];
        float average;
        char grade;
}

int main
{
        examStats aStudent;
```

ตัวแปร aStudent เป็นโครงสร้างชนิด examStats ซึ่งมีสมาชิกรายการแรกเป็น Array ขนาด 3 โดย aStudent.scores[0] จะเป็นการประมวลผลในสมาชิกตัวแรกของ Array สมมุติมีการจัดเก็บข้อมูลในหน่วยความจำดังนี้



เราสามารถสร้าง function ชื่อ printStats เพื่อที่จะพิมพ์ผลลัพท์ออกทางจอภาพได้ดังนี้

```
Exam scores for Judy:  55 90 87
Average score:  77.3333
Letter grade:  C
```

ตัวอย่างที่ 6.14 Function printStats

```
void printStats
        (examStats stuExams)              // IN: the struct to be displayed
{
        cout << "Exam scores for " << stuExams. stuName << " : "
        cout << stuExams.scores[0] << ' ' <<stuExams.scores[1]
             << ' ' << stuExams.scores[2] << endl;
        cout << "Average score: " << stuExams.average << endl;
        cout << "Letter grade  : " << stuExams.grade  << andl;
}
```

ฟังก์ชันนี้ แสดงผลสอบของนักศึกษาออกทางจอภาพโดยมีการเรียกใช้ดังนี้

```
printStats (aStudent);
```

โดยภายใน Function มีการกำหนด formal parameter ที่ชื่อ stuExams ซึ่งมีโครงสร้างเป็น examStats เหมือนกับตัวแปร aStudent ซึ่งเป็นการส่งผ่านค่าแบบ PASS BY VALUE โดยภายใน Function จะจองเนื้อที่เพื่อเก็บข้อมูลที่ส่งมาเพียงชั่วคราวเมื่อประมวลผลเสร็จจะคืนเนื้อที่นี้ให้กับหน่วยความจำส่วนกลางเพื่อใช้งานต่อไป

การทำงานบางครั้งต้องการเปลี่ยนแปลงค่าของข้อมูลเมื่อถูกปฏิบัติงานใน Function สำหรับข้อมูลชนิด Struct เราสามารถส่งผ่านแบบ PASS BY REFERENCE ได้ โดยใช้เครื่องหมาย (&) ระบุดังตัวอย่างที่ 6.15

ตัวอย่างที่ 6.15 Function readEmployee

```
// File: readEmp.cpp
// Reads one employee record into oneEmployee
# include <string>
# include <iostream>

// Pre: None
// Post: Data are read into struct oneEmployee

void readEmployee
     (employee&  oneEmployee)              // OUT: The destination for the data read
{
        cout << "Enter a name terminated with the symbol # : ";
        getline (cin, oneEmployee.name, '#');
        cout << "Enter an id number: ";
        cin >> oneEmployee.id;
        cout << "Enter gender (F or M) : ";
        cin >> oneEmployee.gender;
        cout << "Enter number of dependents: ";
        cin  >> oneEmployee.numDepend;
        cout << "Enter hourly rate: ";
        cin >> oneEmployee.rate;
}
```

ฟังก์ชันนี้ทำหน้าที่อ่านระเบียนพนักงาน 1 คน สมมุติมีการเรียกใช้ดังนี้

```
readEmployee(janitor);
```

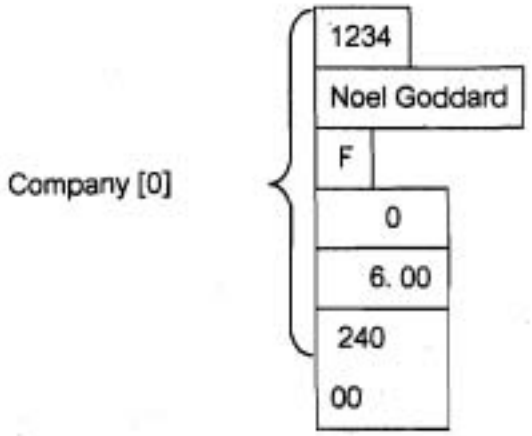
ก่อนการทำงาน janitor ไม่มีข้อมูลใดๆ โดยตัวแปร oneEmployee ในฟังก์ชันจะเป็นตัวแปรที่ใช้อ้างอิงการทำงานโดยใช้เนื้อที่เดียวกันกับ janitor

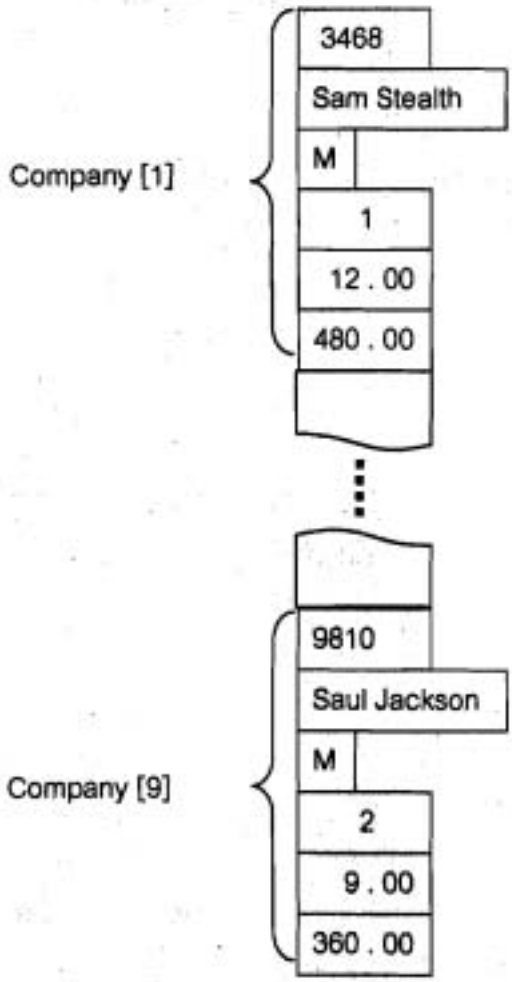
6.7 Array of Structs

ในการแก้ปัญหาโปรแกรมเราสามารถกำหนดโครงสร้างข้อมูลเป็นระเบียนหลายระเบียนโดยติดต่อกัน โดยประกาศเป็นโครงสร้าง Array of Structs ได้ เช่น ต้องการเก็บระเบียนข้อมูลพนักงานจำนวน 10 คน โดยแต่ละคนมีโครงสร้างชนิดเดียวกันชื่อ employee เราสามารถประกาศตัวแปรได้ดังนี้

```
employee company [10];
```

ภาษา C++ จะมีการจัดสรรเนื้อที่เป็น Array ขนาด 10 ช่อง แต่ละช่องมีโครงสร้างเป็นชนิด employee ดังรูปด้านล่างต่อไปนี้ โดยเราใช้ subscript เป็นตัวชี้ข้อมูลในตัวแปร company โดย company [0] จะเป็นการเข้าถึงสมาชิกตัวแรกของ Array เราใช้ตัว notation หรือ (.) ในการเข้าถึง field ต่างๆ ในโครงสร้าง ตัวอย่างเช่น company[9].name เป็นการเข้าถึง field ที่ชื่อ name ของพนักงานคนสุดท้าย





ตัวอย่างที่ 6.16 FUNCTION readCompany

ตัวอย่างนี้เป็น Function ในการอ่านระเบียน พนักงาน 10 คน เก็บใน Array ชื่อ company

```
// File: readCompany.cpp
// Reads 10 employee records into array company
// Pre:  None
//Post: Data are read into array company

void readCompany
  (employee company [ ] )      // OUT: array being read
{
```

```
  // Read 10 employees.
  for (int  i = 0; i < 10;  i++)
  {
    cout << "Enter data for employee " << i+1 << " : "  << endl;
    readEmployee(company[i] ) ;
  }
}
```

กรณีศึกษา : การจองห้องพัก

ปัญหา ต้องการเขียนโปรแกรมภาษา C++ ในการจองห้องพักของสถานที่ตากอากาศแห่งหนึ่ง โดยแบ่งเป็นโมดูลการทำงานดังนี้

โดย Initialize เป็นการกำหนดข้อมูลห้องพักว่างเริ่มต้น กำหนดให้มีเพียง 10 ห้อง

Reserve เป็นการจองห้องพัก โดยกำหนดหมายเลขห้องพัก และชื่อผู้จอง

Print แสดงข้อมูลห้องพักทั้งหมด ทุกห้อง

Search เป็นโมดูลการค้นหาห้องพักว่าง

การออกแบบโปรแกรม

เรามีการออกแบบโมดูลเพิ่มเพื่อให้ผู้ใช้ใช้งานได้ง่ายขึ้นโดยออกแบบโมดูลชื่อ Menu เพื่อแสดงสารสนเทศให้ผู้ใช้สามารถเลือกการปฏิบัติงานที่ต้องการได้โดยแสดงผลทางจอภาพ ดังนี้

```
S:Search room empty
R:Reserve
P:Print all room
E:End
Select : ?
```

ผู้ใช้สามารถเลือกการทำงานที่ต้องการโดยทำงานวนรอบรับข้อมูลจนกระทั่งผู้ใช้ป้อนอักขระ 'E' โปรแกรมจะจบการทำงาน นอกจากนี้เรามีการเพิ่มโมดูลที่ชื่อ Searchroom เพื่อใช้

ในการค้นหาห้องพักที่ต้องการจะจองว่าเป็นห้องพักที่มีสถานะว่างหรือไม่ เพื่อจะที่กรอกรายละเอียดข้อมูลการจองของลูกค้าแต่ละรายการ ในโปรแกรมนี้มีการออกแบบโครงสร้างข้อมูลชนิด Struct ชื่อ room ประกอบด้วยรายการข้อมูลที่จำเป็นดังนี้ หมายเลขห้องพัก(id) กำหนดให้เป็นข้อมูลชนิด int,สถานะห้องพัก(empty) กำหนดให้เป็นข้อมูลชนิด bool กรณีที่เป็น true แสดงว่าห้องพักนี้มีสถานะว่าง กรณีที่เป็น false แสดงว่าห้องพักนี้มีผู้จองแล้ว,ชื่อผู้จอง(name) กำหนดให้เป็นข้อมูลชนิด string ,ราคาห้องพัก(cost) กำหนดให้เป็นข้อมูลชนิด double,ดังโปรแกรมต่อไปนี้

```
#include <iostream>
#include <string>
using namespace std;
struct room
{
        int id;
        bool empty;
        string name;
        double cost;
};
void Initialize(room [] );
void Menu(char &);
void Search(const room []);
void Reserve(room []);
void Print(const room []);
int searchroom(const room [] , int);
int main()
{
        room r[10];
        char ch ;
```

```
        Initialize(r);
        do
        {
                Menu(ch);
                switch  (ch)
                {
                case 'S' : case 's' :    Search(r);
                                         break ;
                case 'R' : case 'r' :    Reserve(r);
                                         break;
                case 'P' : case 'p' :    Print(r);
                }
        }while (ch != 'E' && ch != 'e');
        return 0 ;
}

void Initialize(room r[] )
{
        int i;
        for (i=0 ; i<10 ; i++)
        {
                r[i].id = 1000+i ;
                r[i].empty = true;
                r[i].cost = 1500.00;
        }
}
```

```
void Menu(char &ch)
{
        cout << endl;
        cout << endl;
        cout << "MENU" << endl;
        cout << "S:Search room empty" << endl;
        cout << "R:Reserve" << endl;
        cout << "P:Print all room "   << endl;
        cout << "E:End" << endl;
        cout << "Select :" ; cin >> ch ;
        cout << endl;
}
void Search(const room r[])
{
        for(int i=0 ; i<10 ;i++)
                if (r[i].empty==true)
                        cout << r[i].id << " " ;
        cout << endl;
}
void Reserve(room r[])
{
   string customer;
        int number , i ;
        cout << "name : " ;   cin >> customer;
        Search(r);
        cout << " number =" ; cin >> number ;
        i = searchroom(r , number) ;
```

```
        if (i==-1)
                cout << number << " not -found " << endl;
        else if (r[i].empty==false)
                cout << number << " not empty " << endl ;
        else
        {
                r[i].empty = false;
                r[i].name =  customer ;
                cout << " reserve success " << endl ;
        }
}
void Print(const room r[])
{
        double sum = 0.0 ;
        for (int i=0;i<10 ;i++)
        {
                cout << r[i].id << "  " ;
                if (r[i].empty == true)
                        cout << "empty" << endl;
                else
                {
                        cout << "customer name = " << r[i].name << endl;
                        sum = sum + r[i].cost ;
                }
        }
        cout << " total amount = " << sum << endl;
}
```

```
int searchroom (const room r[] , int number)
{
        for (int i=0 ; i<10 ; i++)
        if (r[i].id==number)
                return i ;
        return -1 ;
}
```

กรณีศึกษา : ร้านขายสินค้า

ปัญหา ต้องการเก็บข้อมูลสินค้า ในร้านขายสินค้าแห่งหนึ่ง โดยข้อมูลที่ต้องการเก็บประกอบด้วยสินค้าที่แตกต่างกันทั้งหมด 100 ชนิด แต่ละชนิดต้องการเก็บข้อมูลดังนี้

1. รหัสสินค้า ใช้ตัวแปร (id)
2. ชื่อสินค้า ใช้ตัวแปร (name)
3. ราคาขาย ใช้ตัวแปร (cost)
4. จำนวนสินค้าในสต๊อก (instock)

จุดประสงค์ในการออกแบบโปรแกรม

1. นำสินค้า 100 ชนิดไปเก็บในตัวแปร อาร์เรย์ของ struct ชื่อ g
2. สร้างเมนูเลือกการทำงาน
   2.1 ซื้อสินค้าเพิ่มในสต๊อก
   2.2 ขายสินค้าให้ลูกค้า
   2.3 พิมพ์สินค้าคงคลังทั้งหมด
   2.4 หยุดการทำงาน
   โดยเลือก 1/2/3/4

   โดยให้มีการออกแบบหน้าจอในลักษณะดังนี้

```
please choose
    1. buy
    2. sell
    3. print
    4. end
select : ?
```

การทำงานเมื่อผู้ใช้

เลือก 1 โปรแกรมจะให้ผู้ใช้ป้อนรหัสสินค้าที่ต้องการเพิ่ม ระบบจะตรวจสอบรหัสว่าถูกต้องหรือไม่

- กรณีที่ถูกต้องโปรแกรมจะให้ป้อนจำนวนสินค้าที่ต้องการเพิ่ม และจะทำการปรับปรุงจำนวนสินค้าให้เพิ่มขึ้น
- กรณีไม่ถูกต้อง จะมี Message บอกถึงความผิดพลาด

เลือก 2 โปรแกรมจะให้ผู้ใช้ป้อนรหัสสินค้าที่ลูกค้าซื้อ ทำนองเดียวกันระบบจะตรวจสอบความถูกต้องของรหัสสินค้าในกรณีที่ถูกต้องจะให้ผู้ใช้ป้อนจำนวนสินค้าที่ต้องการซื้อ ระบบจะตรวจสอบจำนวนสินค้าที่ลูกค้าต้องการซื้อ กับจำนวนสินค้าที่มีอยู่ในสต๊อกถ้ามีสินค้าเพียงพอจะนำราคาขายมาคูณกับจำนวนเงินเพื่อคิดราคาสินค้าและตัดสต๊อกสินค้า เพื่อให้จำนวนสินค้าที่มีอยู่ในสต๊อกถูกต้อง ในกรณีที่เกิดความผิดพลาด เช่น ป้อนรหัสสินค้าผิด หรือ จำนวนสินค้าในสต๊อกมีไม่เพียงพอกับความต้องการของลูกค้า จะมี Message แจ้งถึงความผิดพลาด

เลือก 3 โปรแกรมจะพิมพ์รายละเอียดของสินค้าทั้งหมดออกทางจอภาพ

กรณีศึกษานี้จะเขียนโปรแกรมใน 2 ลักษณะ เพื่อเปรียบเทียบให้เห็นถึงความแตกต่าง โดยโปรแกรมแรก เป็นการเขียนโปรแกรมเดี่ยว ที่มีฟังก์ชัน main เพียงฟังก์ชันเดียว การประกาศโครงสร้างและตัวแปรจะอยู่ภายในฟังก์ชัน main การทำงานจะเป็น ลักษณะแบบเรียงลำดับจากบนลงล่างดังนี้

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main()
{
        struct good
        {
                int id;
                string name;
                double cost ;
                int instock;
        };
        good  g[3];
        char ch;
        int key , n ;
        int i ;

        for (i=0;i<3 ; i++)
        {
                cout<<"id=";cin>>g[i].id;
                cout<<"name";cin>>g[i].name;
                cout<<"cost=";cin>>g[i].cost;
                cout<<"instock=";cin>>g[i].instock;
        }

        do {
                cout <<" please choose" << endl;
                cout <<"1. buy " << endl;
                cout <<"2. sell " << endl;
                cout <<"3. print " << endl;
```

```
cout <<"4.end" << endl ;
cout << "select : ? " ; cin>> ch;
if (ch=='1')
{
        cout<<"Key=" ; cin >> key;
        for (i=0;i<3;i++)
                if (key==g[i].id)
                        break;
        if (i<3)
        {
                cout << " number of item to add = " ;
                cin >> n ;
                g[i].instock += n ;
        }
        else
                cout <<"not found" << endl;

}
else if (ch == '2')
{
                cout<<"id=" ; cin >> key;
        for (i=0;i<3;i++)
                if (key==g[i].id)
                        break;
        if (i<3)
        {
                cout << " number of item to sell = " ;
```

```
                cin >> n ;
                if (n<g[i].instock)
                {
                        g[i].instock -= n ;
                        cout << "total = " << n * g[i].cost ;
                }
                else
                        cout << "in stock = " << g[i].instock
                           << endl;
            }
            else
                cout <<"not found" << endl;
        }
        else if (ch == '3')
        {
            for (i=0;i<3;i++)
            {
                cout << "id:" <<g[i].id <<endl;
                cout << "name :" <<g[i].name <<endl;
                cout << "cost:" << g[i].cost<<endl; ;
                cout << "instock:"<<g[i].instock<<endl ;
            }
        }
    }while (ch!='4');
    cout << "bye bye " << endl;
    return 0 ;
}
```

โปรแกรมนี้จะเห็นได้ว่าไม่มีความยืดหยุ่น มีส่วนของการทำงานที่ซ้ำๆ กัน เช่น การค้นหาข้อมูล และการทำงานภายในฟังก์ชั่นมีการทำงานหลายวัตถุประสงค์ทำให้ติดตามหรือทำความเข้าใจ หรือแก้ไขข้อผิดพลาดได้ยาก การออกแบบโปรแกรมที่ดีเราควรจะแยกความต้องการออกเป็นโมดูลย่อยๆ เพื่อแยกวัตถุประสงค์ให้ชัดเจน เพื่อง่ายต่อการทำความเข้าใจ และเมื่อเกิดข้อผิดพลาดก็สามารถทราบถึงตำแหน่งที่เกิดความผิดพลาด และสามารถแก้ไขเฉพาะส่วนโดยไม่ไปเกี่ยวข้องกับ โมดูลอื่น ๆ ทำให้เกิดประสิทธิภาพในการพัฒนาโปรแกรมมากขึ้น นอกจากนี้ยังสามารถนำโมดูลไปใช้กับโปรแกรมที่มีการทำงานแบบเดียวกันได้ ทำให้ประหยัดในเรื่องของเวลาและค่าใช้จ่ายในการพัฒนาโปรแกรม

```
#include <iostream>
#include <string>
using namespace std;

struct good
        {
                int id;
                string name;
                double cost ;
                int instock;
        };

void input1(good []);
int search(const good g[] , int key);
void menu() ;
void process(good []);
void print(const  good []);
void buy(good []);
void sell(good []);
```

```
int main()
{
        good  g[3];
        input1(g);
        process(g);
        cout << "bye bye " << endl;

   return 0;
}

int search(const good g[] , int key)
{
        for (int i=0;i<3;i++)
                        if (key==g[i].id)
                                        break;
        return i ;
}

void menu()
{
                cout <<" please choose" << endl;
                cout <<"1. buy " << endl;
                cout <<"2. sell " << endl;
                cout <<"3. print " << endl;
                cout <<"4.end" << endl ;
}
```

```
void process(good g[])
{
        char ch;
        do {
                menu();
                cout << "select : ? " ; cin>> ch;
                if (ch=='1')
                        buy(g);
                else if (ch == '2')
                        sell(g);
                else if (ch == '3')
                        print(g);
        }while (ch!='4');
}

void input1(good g[])
{
        for (int i=0;i<3 ; i++)
        {
                cout<<"id=";cin>>g[i].id;
                cout<<"name";cin>>g[i].name;
                cout<<"cost=";cin>>g[i].cost;
                cout<<"instock=";cin>>g[i].instock;
        }
}
```

```
void print(const good g[])
{
        for (int i=0;i<3;i++)
                {
                        cout << "id:" <<g[i].id <<endl;
                        cout << "name :" <<g[i].name <<endl;
                        cout << "cost:" << g[i].cost<<endl; ;
                        cout << "instock:"<<g[i].instock<<endl ;
                }
}

void buy(good g[])
{
        int key ,n , i;

        cout<<"Key=" ; cin >> key;
        i = search(g,key) ;
        if (i<3)
        {
                cout << " number of item to add = " ;
                cin >> n ;
                g[i].instock += n ;
        }
        else
                cout <<"not found" << endl;
}
```

```
void sell(good g[])
{
        int key ,n , i;

        cout<<"id=" ; cin >> key;
        i = search(g,key);
        if (i<3)
        {
                cout << " number of item to sell = " ;
                cin >> n ;
                if (n<g[i].instock)
                {
                        g[i].instock -= n ;
                        cout << "total = " << n * g[i].cost ;
                }
                else
                        cout << "in stock = " << g[i].instock << endl;
        } .
        else
                cout <<"not found" << endl;
}
```

การส่งผ่านข้อมูลชนิด Struct ระหว่าง Module เรามีความจำเป็นต้องกำหนดโครงสร้างข้อมูลไว้ภายนอกฟังก์ชั่น main เพื่อให้ทุก Module ในโปรแกรมรู้จัก และสามารถอ้างถึงได้ เพราะถ้ากำหนดไว้ภายในฟังก์ชั่น main จะคล้ายกับเป็นประกาศโครงสร้างที่ใช้เฉพาะในฟังก์ชั่น main เท่านั้น

สรุป

1. ตัวแปรชนิดอาเรย์นั้นจะมีการนำข้อมูลที่มีชนิดเดียวกันมาเก็บรวมกัน ซึ่งการอ้างข้อมูลต่างๆที่เก็บในอาเรย์นั้นอาจเป็นชนิดหนึ่งมิติ(one-dimension) สองมิติหรือหลายมิติ (two-more dimension)
2. รูปแบบของการประกาศตัวแปรหนึ่งมิติ

   dataType arrayName[intExp] ;

   เช่น int num[5] ; การประกาศตัวแปรนี้จะส่งผลให้ตัวแปร num สามารถเก็บข้อมูลได้ 5 จำนวนเป็นเลขจำนวนเต็ม อันประกอบด้วยตัวแปร num[0] ,num[1] ,num[2] ,num[3] ,num[4]
3. ค่าของข้อมูลที่เก็บในตัวแปรชนิดอาเรย์มาใช้งานได้เพียงแต่ระบุชื่อตัวแปรและลำดับหรือ ตัวชี้ (index) ข้อมูลให้ถูกต้อง เรานิยมใช้คำสั่ง for มาใช้ในกำหนดหรือเก็บข้อมูลหรือการเข้าถึงข้อมูลในตัวแปรอาเรย์ในลำดับต่างๆ โดยให้ตัวแปรที่เป็นตัวนับเริ่มจาก 0 จนถึง ลำดับสุดท้าย
4. เราสามารถส่งอาเรย์ผ่านฟังก์ชันได้ โดยเขียนชื่อเฉพาะตัวแปรแต่ไม่ต้องมี subscript ของการเรียกใช่ฟังก์ชัน สำหรับส่วนของ formal parameter list ของฟังก์ชันต้องมีการกำหนดพารามิเตอร์ให้สอดคล้องกับอากิวเมนต์ที่ส่งผ่านค่าโดยมีชนิดข้อมูลเหมือนกัน และมีการกำหนด subscript เพื่อใช้อ้างถึงสมาชิกในอาเรย์อาจระบุขนาดหรือไม่ระบุขนาดก็ได้
5. ภาษา C++ นั้นยินยอมให้มีการส่งผ่านโครงสร้างข้อมูลชนิดอาเรย์แบบอ้างอิง โดยไม่มีการสร้างเนื้อที่ขึ้นใหม่แต่อ้างอิงใช้เนื้อที่ที่มีการเรียกใช้ที่สอดคล้องกันคำเฉพาะ const ที่กำหนดหน้าพารามิเตอร์ แสดงให้ทราบว่าพารามิเตอร์นั้นไม่สามารถเปลี่ยนภายในฟังก์ชันได้ สำหรับการเรียกใช้ (function call) เราเขียนชื่อ และอากิวเมนต์จริงโดยไม่ต้องใส่ [] หลังชื่อของตัวแปรอาเรย์
6. การกำหนดตำแหน่งของข้อมูลในลักษณะ 2 มิติ กล่าวคือมีตัวชี้ 2 ตัว มีรูปแบบดังนี้

   datatype arrayName[intExp1][intExp2] ;

โดย intExp1 และ intExp2 ต้องเป็นค่าคงที่ที่มีค่าเป็นเลขจำนวนเต็มบวกเท่านั้น ทำหน้าที่เป็นตัวชี้ของข้อมูล โดยเปรียบ intExp1 เป็นแถว และ intExp2 เป็นคอลัมภ์ เช่น float matrix[2][3]; เครื่องคอมพิวเตอร์จะจัดสรรเนื้อที่ว่างสำหรับเก็บค่าที่เป็นทศนิยม 2x3 = 6 ช่อง

7. ภาษา C++ นั้นเราสามารถกำหนด Array ได้หลายมิติ เช่น กำหนด Array ที่ชื่อว่า sales ดังนี้

```
const int PEOPLE = 10;
const int YEARS =5;
double sales[PEOPLE] [YEARS] [12];
```

จากตัวอย่างที่กำหนดข้างต้น ตัวแปร sales จะถูกจัดสรรข้อมูล 600 ช่อง ต่อเนื่องกันไป ซึ่งได้จากการนำ PEOPLE*YEARS*12 ก็คือ 10*5*12 มีค่า = 600 นั่นเอง โดยsales [0] [2] [11] จะแทนการขายของพนักงานขายคนที่ 1 ซึ่งมี subscript เป็น 0 ในช่วงของเดือนธันวาคม ซึ่งมี subscript เป็น 11 ของปีที่ 3 ซึ่งมี subscript เป็น 2

8. รูปแบบการประกาศโครงสร้างข้อมูลชนิด struct

```
struct struct-type
{
   Type1 id-list1;
   Type2 id-list2;

   type n id-list n;
};
```

เป็นการกำหนดโครงสร้างชนิดข้อมูลชนิด Struct โดย Sturct-type คือชื่อของโครงสร้างที่ผู้ใช้กำหนดขึ้น id-list คือ รายการข้อมูลในโครงสร้าง

9. การกำหนดโครงสร้างชนิด struct นั้นระบบยังไม่มีการจัดสรรเนื้อที่แต่อย่างใด เราสามารถจัดสรรเนื้อที่โดยกำหนดตัวแปรใหม่เพื่อเก็บข้อมูลในโครงสร้างนี้ได้ ดังนี้

```
employee organist, janitor;
```

เราสามารถเข้าถึงสมาชิกของโครงสร้างโดยอ้างสมาชิกของตัวแปรโดยใช้เครื่อง (.) ดังนี้

```
organist.id = 1234;
organist.name = "Noel Goddard";
```

10. การส่งผ่านข้อมูลชนิด Struct ไปยัง function ชนิดต่างๆ โดยต้องจัดเตรียม actual Arguments ให้มีชนิดเดียวกันกับ formal parameter การทำงานบางครั้งต้องการเปลี่ยนแปลงค่าของข้อมูลเมื่อถูกปฏิบัติงานใน Function สำหรับข้อมูลชนิด Struct เราสามารถส่งผ่านแบบ PASS BY REFERENCE ได้ โดยใช้เครื่องหมาย (&)

11. ในการแก้ปัญหาโปรแกรมเราสามารถกำหนดโครงสร้างข้อมูลเป็นระเบียนหลายระเบียนติดต่อกัน โดยประกาศเป็นโครงสร้าง Array of Structs ได้ เช่น ต้องการเก็บ ระเบียนข้อมูลพนักงานจำนวน 10 คน โดยแต่ละคนมีโครงสร้างชนิดเดียวกันชื่อ employee เราสามารถประกาศตัวแปรได้ดังนี้

```
employee company [10];
```

ภาษา C++ จะมีการจัดสรรเนื้อที่เป็น Array ขนาด 10 ช่อง แต่ละช่องมีโครงสร้างเป็นชนิด employee โดยเราใช้ subscript เป็นตัวชี้ข้อมูลในตัวแปร company โดย company [0] จะเป็นการเข้าถึงสมาชิกตัวแรกของ Array เราใช้ตัว notation หรือ (.) ในการเข้าถึง field ต่างๆ ในโครงสร้าง ตัวอย่างเช่น company[9].name เป็นการเข้าถึง field ที่ชื่อ name ของพนักงานคนสุดท้าย

แบบฝึกหัด

1. จงอธิบายความแตกต่างระหว่าง ตัวแปร X5 กับ X[5]
2. จงอธิบายความแตกต่างระหว่าง int score[10] กับ char grade[10]
3. จงอธิบายถึงการประกาศ Array ดังต่อไปนี้

   3.1 int workers[3] = {6,8,8,0}:

   3.2 int freq[12] = {0,0,3,7,10,16,28,31}:

4. พิจารณาส่วนของคำสั่งต่อไปนี้แล้วตอบคำถาม

   4.1
```
for (int  i=9;i>=0; i- -)
    cout<< x[i]<<"  ";
cout<< endl;
```
   ผลของการทำงานในคำสั่งข้างต้นเป็นอย่างไร

   4.2
```
i = 0;
while (i<10)
{
        cout<<x[i]<<"  ";
        i+=2;
}
cout<<endl;
```
   ผลของการทำงานในคำสั่งข้างต้นเป็นอย่างไร

   4.3
```
for (i=0;i<10;i++)
  for(j=0;j<10;j++)
        cout<<i<<'  '<<j;
```
   ผลของการทำงานในคำสั่งข้างต้นเป็นอย่างไร

   4.4
```
#include <iostream>
using namespace std;
int main()
```

```
{
  int count;
  int alpha [5];
  alpha [0] = 5;
  for (count = 1; count <5; count++)
  {
      alpha[count] = 5 * count + 10;
      alpha[count – 1] = alpha [count] – 4;
  }
  cout<< "List elements: ";
  for  (count = 0; count < 5;count ++)
       cout<< alpha[count]<< "  ";
  cout << endl;
return 0;
}
```

ผลลัพธ์การทำงานของโปรแกรมข้างต้นนี้เป็นอย่างไร

4.5
```
#include<iostream>
using namespace std;
int main()
{
    int j;
    int one[5];
    int two[10];
    for (j = 0; j<5; j++)
      one[ j ] = 5 * j + 3;
    cout<< "One Contains: ";
    for ( j = 0; j<5; j++)
```

```
            cout << one[ j ] << " ";
    cout<< endl;
    for ( j = 0; j<5; j++)
    {
        two[ j ] = 2 * one[ j ] – 1;
        two[ j+5] = one[4-j ] + two[ j ] ;
    }
    cout<<" Two Contains: ";
    for ( j = 0; j<10; j++)
       cout<< two[ j ] <<"  ";
    cout << endl;
    return 0;
  }
```

ผลลัพธ์การทำงานของโปรแกรมข้างต้นเป็นอย่างไร

4.6 สมมุติมีการประกาศตัวแปรอาร์เรย์ 2 มิติ ดังนี้

```
int beta [ 3 ] [3 ];
for ( i = 0; i<3; i++)
    for ( j = 0; j<3; j++)
       beta [ i ]  [ j ] = 2 * ( i + j ) % 4;
```

จงแสดงค่าที่เก็บในตัวแปร beta หลังจากประมวลคำสั่งข้างต้น

5. จงเขียนโปรแกรมในการตรวจสอบการจองตั๋วที่นั่งของสายการบินแห่งหนึ่ง ซึ่งประกอบด้วย 13 แถวๆ ละ 6 ที่นั่ง โดยแถวที่ 1 และ 2 เป็นที่ชั้น First Class แถวที่เหลือเป็น Economy Class และแถวที่ 1 ถึง 7 เป็นแถวที่ห้ามสูบบุหรี่ การทำงานของโปรแกรมจะให้ผู้ใช้ป้อนสารสนเทศดังนี้

- ชนิดของตั๋ว(First Class หรือ Economy Class)
- ถ้าเป็น Economy Class ต้องบอกด้วยว่า สูบบุหรี่หรือไม่สูบบุหรี่
- ที่นั่งที่จอง

- ผลลัพธ์ของการจองจะอยู่ในรูปแบบดังนี้

	A	B	C	D	E	F
Row 1	*	*	X	*	X	X
Row 2	*	X	*	X	*	X
Row 3	*	*	X	X	*	X
Row 4	X	*	X	*	X	X
Row 5	*	X	*	*	X	X
Row 6	*	X	*	*	*	X
Row 7	X	*	*	*	X	X
Row 8	*	X	*	X	X	*
Row 9	X	*	X	X	*	X
Row 10	*	X	*	X	X	X
Row 11	*	*	X	*	X	*
Row 12	*	*	X	X	*	X
Row 13	*	*	*	*	X	*

โดยเครื่องหมาย "*" แสดงถึงที่นั่งที่ว่าง ส่วน X แสดงถึงที่นั่งที่ถูกจองแล้ว

6. จงเขียนโปรแกรมในการสร้าง Array 1 มิติ ขนาด 16 ช่อง โดยให้มีการเพิ่มขึ้นเป็นลำดับ โดยกำหนดค่าเริ่มต้น First และผลต่างคือ Diff ดังตัวอย่างต่อไปนี้

First=21 และ Diff=5 ค่าที่เก็บใน Array ทั้ง 16 ช่อง จะเก็บเรียงลำดับดังนี้

21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96.

โดยเขียนเป็นฟังก์ชั่นที่ชื่อ CreateArithmeticSequence

6.1 เขียนฟังก์ชั่นชื่อ MetrixSize ในการอ่านข้อมูลจาก Array 1 มิติที่สร้างไว้มาเก็บใน Array 2 มิติ ขนาด 4 แถว 4 คอลัมภ์ ผลจากการทำงานจะเก็บเป็น Array 2 มิติ ที่มีข้อมูลดังนี้

21 26 31 36

41 46 51 56

61 66 71 76

81 86 91 96

6.2 เขียนฟังก์ชั่นชื่อ Reversediagonal ในการสลับค่าของ เส้นทแยงมุมของ Array 2 มิติ ที่สร้างขึ้นในข้อ 6.1 ผลจากการทำงาน จะมีการเก็บข้อมูลดังนี้

96 26 31 81

41 71 66 56

61 51 46 76

36 86 91 21

6.3 เขียนฟังก์ชั่น PrintMetrix ในการแสดงผลลัพธ์ของสมาชิกของตัวแปร Array 2 มิติ ในรูปแบบของ 4 แถว 4 คอลัมภ์

7. จงเขียนโปรแกรมรับเลขจำนวนเต็ม 10 จำนวน ใดๆ เก็บในตัวแปร Array 1 มิติ จง แสดงผลข้อมูลทั้งหมดที่เป็นเลขจำนวนเต็มคี่ และผลรวมเฉพาะเลขจำนวนเต็มคี่ทั้งหมดออกทางจอภาพ
8. ให้รับข้อมูลทางแป้นพิมพ์ และแสดงส่วนกลับของข้อความออกทางจอภาพ
9. ให้เขียนฟังก์ชั่นในการนับจำนวนของสมาชิกใน Array ที่เป็นเลขจำนวนเต็มที่มีค่ามากกว่า 0
10. จงเขียนฟังก์ชั่นที่ชื่อว่า linSearchLast เพื่อหาค่าของข้อมูลตัวสุดท้ายของArray 1 มิติ และ 2 มิติใดๆ
11. จงเขียนฟังก์ชั่นในการอ่านจุด coordinate บนระนาบ x และ y โดยกำหนดโครงสร้างจุดเป็นดังนี้

```
struct point
{
        float x;
        float y;
};
```

การทำงานของฟังก์ชั่นนี้จะมีการส่งผ่านค่า ผ่านตัวแปร a ซึ่งเป็นอาร์กิวเมนต์ที่มีชนิดข้อมูลเป็น point

12. จงเขียนโปรแกรมในการอ่านข้อมูลไปเก็บไว้ใน 2 Array ที่มีขนาด 20 ช่อง ของตัวแปร

Array x และ y ตามลำดับ จงสร้าง Array z ซึ่งสมาชิกของ z เกิดจากการนำผลรวมของ x และ y มาหารากที่สอง ดังตัวอย่างต่อไปนี้

	0	1	2	.....		19
X	6	20	.....	.....	.....	80

	0	1	2	.....		19
y	30	5	...	.....	.....	1

	0	1	2	.....		19
z	6.0	5.0	....	.....	.....	9.0

13. จงเขียนโปรแกรมหาจำนวนเฉพาะของตัวเลขจำนวนเต็มที่มีค่าน้อยกว่า 2000 โดยสร้าง Array ของค่า Boolean เพื่อใช้ตรวจสอบข้อมูลในแต่ละค่า โดยถ้าค่าใดเป็นจำนวนเฉพาะจะมีการเก็บค่า True มิฉะนั้นแล้วก็จะเก็บค่า False กำหนดให้ค่าเริ่มต้นของทุกช่องมีค่าเป็น True
14. จงเขียนโปรแกรมในการสร้าง Morse code จากประโยคที่ต้องการโดยตัวอักษรที่เกี่ยวพันธ์กับรหัสเป็นดังนี้

ตัวอักขระ	Morse code
A	C
B	D
..	
..	
Y	A
Z	B

สมมุติผู้ใช้ป้อนข้อมูล BOY โปรแกรมจะแปลงข้อมูลเป็นรหัส Morse ได้ผลลัพธ์คือ DQA กำหนดให้โปรแกรมมีฟังก์ชั่นดังนี้

void readCode(string codeArray[]);

void writeCode(string codeArray[]);

15. จงเขียนโปรแกรมเกมส์ Hang Man ในการเดาคำภาษาอังกฤษ โดยถ้าผู้เล่นทายตัวอักษรที่มีอยู่ในคำผิดเกิน 5 ครั้ง โปรแกรมจะหยุดการทำงานและเฉลยคำที่ถูกต้องออกทางจอภาพ
16. จงเขียนโปรแกรมในการรวมสมาชิกที่เรียงลำดับที่เก็บอยู่ใน 2 อาร์เรย์ ใดๆ โดยไม่มีข้อมูลที่ซ้ำกัน(Merge Sort)
17. จงออกแบบโปรแกรมในการจองห้องพัก ซึ่งโรงแรมมีขนาดห้องพักทั้งหมด 150 ห้อง แบ่งเป็นห้อง SUIT เตียงเดี่ยวและคู่,SUPLIER เตียงเดี่ยวและคู่,DELUXE(sea view) เตียงเดี่ยวและคู่,STANDARD(garden view) เตียงเดี่ยวและคู่ ในการจองห้องพักให้จองได้เพียง 2 ห้องเท่านั้น และต้องทำการชำระเงินจองเป็นจำนวนเงิน 30% ของราคาห้องพัก และต้องชำระเงินก่อนล่วงหน้า2สัปดาห์ และข้อมูลลูกค้าที่ต้องการบันทึกคือ IDcard,Name,Address,Telephone, และโปรแกรมสามารถหายอดการจองของห้องพักในแต่ละวัน และสามารถคำนวณหารายได้ทั้งเดือนของโรงแรม