บทที่ 9
ข้อมูล structure และ union

♦ ความหมายของ structure

structure หมายถึง กลุ่มข้อมูลซึ่งประกอบด้วยข้อมูลชนิดเดียวกันหรือหลายชนิดก็ได้ ข้อมูลแต่ละรายการใน structure เรียกว่า สมาชิกของสตรักเจอร์ (members of structure) ***structure แตกต่างจากอาร์เรย์*** คือ สมาชิกของ structure แต่ละรายการเป็น ***ข้อมูลต่างชนิดกันได้*** แต่สมาชิกของอาร์เรย์แต่ละรายการจะต้องเป็นชนิดเดียวกันเท่านั้น ตัวอย่างข้อมูลที่สามารถนำมาเก็บเป็นชนิด structure ได้แก่ข้อมูลที่มีรายละเอียดภายในแต่ละรายการเป็นข้อมูลต่างชนิด(Data Type) กัน เช่น

- ข้อมูลประวัตินักศึกษา ประกอบด้วย รหัสนักศึกษา, ชื่อนักศึกษา, อายุ, วันเดือนปีเกิด
- ข้อมูลรายละเอียดวิชา ประกอบด้วย รหัสวิชา, ชื่อวิชา, จำนวนหน่วยกิต, ผู้สอน
- ข้อมูลประวัติสินค้า ประกอบด้วย รหัสสินค้า, คำอธิบายรายการ, หน่วยนับ, ราคาต่อหน่วย

♦ วิธีสร้างข้อมูลประเภท structure และการนำไปใช้

ก่อนที่จะนำข้อมูลประเภท structure มาใช้ในโปรแกรม ***จะมี 2 ขั้นตอน*** คือ

1. สร้างชื่อประเภทข้อมูลให้เป็นประเภท structure ก่อนโดยผู้ใช้กำหนดชื่อ structure จาก keyword คำว่า **struct**

2. จากนั้นจากนำ **ชื่อประเภทข้อมูล structure** ไปประกาศตัวแปรที่ต้องการสร้างขึ้นใช้ภายในโปรแกรม

การสร้างประเภทข้อมูลให้เป็น structure มีรูปแบบโครงสร้าง ดังนี้

```
struct  ชื่อสตรักเจอร์
        {
          ชนิดข้อมูล  ชื่อรายการ;
          ชนิดข้อมูล  ชื่อรายการ;
          ชนิดข้อมูล  ชื่อรายการ;
          ชนิดข้อมูล  ชื่อรายการ;
      } ;
```

เช่น ถ้าต้องการสร้างข้อมูลแบบ structure เพื่อเก็บข้อมูล รหัสนักศึกษา, ชื่อนักศึกษา, คะแนนสอบระหว่างภาค, คะแนนสอบปลายภาค และเกรด กำหนดรายละเอียดใน structure ดังนี้

```
struct STD
        { char   code[9];
          char   name[31];
          int    midterm;
          int    final;
          char   grade;
        };
```

ชื่อ structure กำหนดโดยผู้ใช้

ชื่อ ฟิลด์(Field) ในโครงสร้าง

STD คือชื่อของประเภท **structure** ที่มีสมาชิกหรือฟิลด์(Field) 5 รายการ คือ code, name, midterm, final และ grade โดยที่สมาชิกแต่ละรายการมีชนิดข้อมูล***ที่แตกต่างกันได้***

การประกาศตัวแปร

เมื่อต้องการนำชื่อ STD ไปใช้ประกาศตัวแปร สามารถประกาศตัวแปรได้ตามรูปแบบการประกาศตัวแปรประเภทอื่น ๆ ทำได้ 2 วิธี คือ

<u>วิธีแรก</u> ประกาศตัวแปรไปพร้อมกับการสร้าง structure

```
struct STD
        { char   code[9];
          char   name[31];
          int    midterm;
          int    final;
          char   grade;
        } student;
```

<u>วิธีแรก</u> ประกาศตัวแปร หลังจากสร้าง structure แล้ว

```
struct STD
        { char   code[9];
          char   name[31];
          int    midterm;
          int    final;
          char   grade;
        };

STD student;
```

ชื่อตัวแปร Student เป็นประเภท STD

♦ การอ้างถึง structure และสมาชิกของ structure

การอ้างถึงสมาชิกใน structure ไม่สามารถจะอ้างถึงชื่อรายละเอียดใน structure โดยตรงได้ **แต่จะต้องกำหนดตัวแปร(variable) ที่มีชนิดเป็น structure ขึ้นมาก่อน** เช่น จากตัวอย่างที่สร้าง structure ชื่อ STD ขึ้น เราสามารถประกาศตัวแปร student และ stud ให้เป็นประเภท STD ได้ดังนี้

STD **student,stud; //ชื่อตัวแปรคือ student และ stud**

หรืออาจประกาศในขณะสร้าง structure ก็ได้ มีรูปแบบดังนี้

```
struct STD
        {  char   code[9];
           char   name[31];
          int      midterm;
           int      final;
           char    grade;
        }student,stud;
```

เมื่อเรากำหนดตัวแปร **student และ stud ให้เป็นชนิด STD** ซึ่งเป็น structure ได้แล้ว เราสามารถอ้างถึงสมาชิกของ **STD** ได้โดยการเขียน***ชื่อตัวแปรเครื่องหมายจุดและตามด้วยชื่อรายการสมาชิกหรือฟิลด์(Field)ของ STD*** ดังนี้

	student.code	หมายถึง รหัสนักศึกษา
	student.name	หมายถึง ชื่อนักศึกษา
	student.midterm	หมายถึง คะแนนสอบระหว่างภาค
	student.final	หมายถึง คะแนนสอบปลายภาค
	student.grade	หมายถึง เกรด
หรือ	stud.code	หมายถึง รหัสนักศึกษา
	stud.code	หมายถึง รหัสนักศึกษา
	stud.name	หมายถึง ชื่อนักศึกษา
	stud.midterm	หมายถึง คะแนนสอบระหว่างภาค
	stud.final	หมายถึง คะแนนสอบปลายภาค
	stud.grade	หมายถึง เกรด

♦ การกำหนดค่าคงที่ให้กับสมาชิกของ structure

```
struct  STD
        {  char   code[9];
           char   name[31];
           int      midterm;
           int      final;
           char    grade;
        }student,stud;
```

จากโครงสร้างของ STD และการกำหนดตัวแปร student , stud สามารถกำหนดค่าให้กับตัวแปรได้ดังนี้

1. กำหนดค่าคงที่ให้กับ structure เขียนได้ดังนี้

```
strcpy(student.code,"40214514");        //เนื่องจากเป็น string ต้องใช้ strcpy()
strcpy(student.name, "Somsak");         //เนื่องจากเป็น string ต้องใช้ strcpy()
student.midterm = 45;
student.final = 30;
student.grade=75;
```

2. การกำหนดค่าของตัวแปรให้กับ structure เขียนได้ดังนี้

```
char co[9]=”10015210”;
strcpy(student.code,co);
int  x = 25;
student.midterm = x;
```

♦ การรับข้อมูลจากคีย์บอร์ดเก็บในสมาชิกของ sturcture

การรับค่าคงที่จากคีย์บอร์ดไปเก็บไว้ใน structure ทำได้โดยใช้ cin>> ดังนี้

```
cin>>student.code;
cin>>student.name;
cin>>student.midterm;
cin>>student.final;
cin>>student.grade;
```

♦ การแสดงข้อมูลจากสมาชิกของ structure

การแสดงผลข้อมูลจาก structure ทำได้โดยใช้ cout<< ดังนี้

```
cout<<student.code;
cout<<student.name;
cout<<student.midterm;
cout<<student.final;
cout<<student.grade;
```

- ***ตัวอย่างโปรแกรม stru_exp1.cpp แสดงการกำหนดโครงสร้าง structure การประกาศตัวแปร การรับค่าทางคีย์บอร์ด และแสดงค่าใน structure เพื่อเก็บรายละเอียดข้อมูลคะแนนสอบนักศึกษา จำนวน 1 คน***

```
/*Program : stru_ex1.cpp
 Process : creat structure and manage structure*/

 #include <iostream.h>
 #include <conio.h>
  //declaration structure STD
  struct STD
     { char  code[9];
       char  name[31];
       int   midterm;
       int   final;
       char  grade;
     };
 //declaration prototype function
 void input();
 void display();
 //declaration student is global variable has structure type
 STD student;
void main() //begin main program
 {
  clrscr();
  input(); clrscr();
  display();
  getch();
 }  //end main program


 void input()  //function enter data
```

```
{ clrscr();
  gotoxy(30,2);cout<< " Please enter data : ";
  gotoxy(5,4);cout<< "Code : ";
  gotoxy(5,5);cout<< "Name : ";
  gotoxy(5,6);cout<< "Midterm : ";
  gotoxy(5,7);cout<< "Final : ";
  gotoxy(5,8);cout<< "Grade : ";
  //input from keyboard
  gotoxy(15,4);cin>>student.code;
  gotoxy(15,5);cin>>student.name;
  gotoxy(15,6);cin>>student.midterm;
  gotoxy(15,7);cin>>student.final;
  gotoxy(15,8);cin>>student.grade;
}

void display()  //function display data
{
  gotoxy(30,2);cout<< " Your Information  : \a";
  gotoxy(5,4);cout<< "Code : "<<student.code;
  gotoxy(5,5);cout<< "Name : "<<student.name;
  gotoxy(5,6);cout<< "Midterm : "<<student.midterm;
  gotoxy(5,7);cout<< "Final : "<<student.final;
  gotoxy(5,8);cout<< "Grade : "<<student.grade;
}
```

♦ อาร์เรย์ของ structure

การกำหนดอาร์เรย์ของ structure มีวิธีการเช่นเดียวกับการกำหนดอาร์เรย์ 1 มิติปกติ เพราะถือว่า ***structure เป็นข้อมูล 1 รายการในอาร์เรย์*** ดังนั้นการอ้างอิงถึงข้อมูลที่เป็นอาร์เรย์ชนิดโครงสร้าง ***จึงต้องอ้างถึงตำแหน่งรายการของสมาชิกในอาร์เรย์ร่วมกับการอ้างถึงสมาชิกใน structure***

```
struct STD              //สร้าง structure ชื่อ STD
        {  char  code[9];
           char  name[31];
           int   midterm;
           int   final;
           char  grade;
        };
STD student[5];
```

//ประกาศตัวแปรชื่อ student เป็นอาร์เรย์มีสมาชิกเป็น structure ชื่อ STD จำนวน 5 รายการ

ฟิลด์(Field) หรือสมาชิกของโครงสร้าง

index ของ array

	code[9]	name[31]	midterm	final	grade
0	10000001	somsak	55	20	B
1	...	...	...	...	...
2					
3					
4					

ดังนั้น ***การอ้างถึงสมาชิกของอาเรย์ต้องใช้ [index] ตามหลังชื่อตัวแปรและอ้างถึงสมาชิกของ structure ต้องใช้ ตามด้วยจุด (.) และชื่อรายการสมาชิกใน structure*** ดังนี้

```
student[0].midterm=30;
student[1].final=50;
strcpy(student[0].code,"10012512");
cin>>student[4].code;
cout<<student[4].code;
```

- ***ตัวอย่างโปรแกรม stru_exp1.cpp แสดงการใช้โครงสร้างข้อมูลอาร์เรย์ร่วมกับโครงสร้างข้อมูลแบบ structure เพื่อเก็บรายละเอียดข้อมูล รหัส, ชื่อ, คะแนนสอบระหว่างภาค, ปลายภาค และเกรดของนักศึกษา จำนวน 5 คน***

```
/*Program : stru_exp1.cpp
  Process : creat structure and manage structure*/

  #include <iostream.h>
  #include <conio.h>

   //declaration structure STD
   struct STD
      { char  code[9];
        char  name[31];
        int   midterm;
        int   final;
        char  grade;
      };
  //declaration prototype function
  void input();
  void display();
  //declaration array structure is global variable
  STD student[5];

void main() //begin main program
  {
    clrscr();
    input(); clrscr();
    display();
  }  //end main program

  void input()  //function enter data
  { int i;
    for(i=0;i<=4;++i)
    {  clrscr();
       gotoxy(15,2);cout<< " Please enter data of student No.# \a"<<i+1;
       gotoxy(5,4);cout<< "Code : ";
       gotoxy(5,5);cout<< "Name : ";
       gotoxy(5,6);cout<< "Midterm : ";
       gotoxy(5,7);cout<< "Final : ";
       gotoxy(5,8);cout<< "Grade : ";
       gotoxy(15,4);cin>>student[i].code;
```

```
    gotoxy(15,5);cin>>student[i].name;
    gotoxy(15,6);cin>>student[i].midterm;
    gotoxy(15,7);cin>>student[i].final;
    gotoxy(15,8);cin>>student[i].grade;
   }
 }

 void display()  //function display data
 {int i;
  for(i=0;i<=4;++i)
   {  clrscr();
    gotoxy(30,2);cout<< " Your Information  : studen#\a"<<i+1;
    gotoxy(5,4);cout<< "Code : "<<student[i].code;
    gotoxy(5,5);cout<< "Name : "<<student[i].name;
    gotoxy(5,6);cout<< "Midterm : "<<student[i].midterm;
    gotoxy(5,7);cout<< "Final : "<<student[i].final;
    gotoxy(5,8);cout<< "Grade : "<<student[i].grade;
    cout<<endl<<endl<< "press any key to continue..."; getch();
   }
 }
```

♦ ข้อมูลชนิด union

ยูเนียน(union) เป็นกลุ่มข้อมูลที่มีโครงสร้างเช่นเดียวกับ structure ***แต่มีความแตกต่างกันในด้านวิธีการกำหนด และวิธีการจัดเก็บข้อมูลในหน่วยความจำ***

การจัดเก็บข้อมูลในหน่วยความจำ สมาชิกแต่ละรายการของ structure จะจองพื้นที่จัดเก็บข้อมูลในหน่วยความจำอิสระจากกัน แต่ union สมาชิกทุกตัวจะจองพื้นที่ในหน่วยความจำตำแหน่งเดียวกัน และใช้หน่วยความจำตำแหน่งนั้นร่วมกัน สมาชิกแต่ละตัวจะเข้าไปใช้หน่วยความจำของพร้อมกันไม่ได้ จะต้องสลับกันใช้หน่วยความจำตำแหน่งนั้น จึงต้องระมัดระวังความถูกต้องของข้อมูลเมื่อจะอ้างถึงสมาชิกแต่ละตัวใน union มีประโยชน์ในเรื่องของการประหยัดหน่วยความจำ ถ้าจำเป็นต้องใช้ตัวแปรหลายตัว แต่เข้าไปใช้หน่วยความจำไม่พร้อมกัน

การกำหนดข้อมูลแบบ union มีรูปแบบดังนี้

```
union ชื่อยูเนียน
        { ชนิดข้อมูล ชื่อรายการสมาชิก;
          ชนิดข้อมูล ชื่อรายการสมาชิก;
          ชนิดข้อมูล ชื่อรายการสมาชิก;
          ชนิดข้อมูล ชื่อรายการสมาชิก;
        };
```

เช่น

```
union STD2
      {  char   code[9];
         char   name[31];
         int    midterm;
         int    final;
         char   grade;
      };
```

- ***ตัวอย่างโปรแกรม union.cpp แสดงการใช้ข้อมูลประเภท structure เปรียบเทียบกับการใช้ union เนื่องจากสมาชิกของ structure ใช้วิธีการเก็บข้อมูลในหน่วยความจำแยกกันเป็นอิสระจึงสามารถเก็บข้อมูลพร้อมกันได้ และนำมาใช้ได้ถูกต้อง ส่วน union สมาชิกทุกรายการใช้หน่วยความจำตำแหน่งเดียวกัน สมาชิกจะเข้ามาใช้พร้อมกันไม่ได้ ดังนั้นจึงแสดงข้อมูลที่เก็บไว้ได้ไม่ถูกต้อง เมื่อกำหนดให้เก็บข้อมูลพร้อมกัน ถูกต้องเฉพาะรายการสุดท้ายที่เข้าไปใช้หน่วยความจำ***

```
/*Program : union.cpp
  Process : creat structure and union ,compare using memory*/

  #include <iostream.h>
  #include <conio.h>
  #include <string.h>
```

```
  //declaration structure STD
  struct STD
     { char  code[9];
       char  name[31];
       int   midterm;
       int   final;
       char  grade;
     };
 //declare union STD2
 union STD2
     { char  code[9];
       char  name[31];
       int   midterm;
       int   final;
       char  grade;
};

 //declaration global variable
 STD  student;  //type is structure
 STD2 stu;      //type is union

 void main() //begin main program
 {
  clrscr();
  //set value of student variabel...struct type
    strcpy(student.code,"100001");
    strcpy(student.name,"Somsak");
    student.midterm=35;
    student.final=30;
    student.grade='D';

  //set value of stu variable... union type
    strcpy(stu.code,"100001");
    strcpy(stu.name,"Somsak");
    stu.midterm=35;
    stu.final=30;
    stu.grade='D';
    //display data from student (struct)
   cout<<  "Display data from student variable of struct"<<endl;
   cout<< "student.code= "<<student.code<<endl;
   cout<< "student.name= "<<student.name<<endl;
   cout<< "student.midterm= "<<student.midterm<<endl;
   cout<< "student.final= "<<student.final<<endl;
```

```
   cout<< "student.grade= "<<student.grade<<endl;
   cout<< "*** correct value of student ***\a"<<endl<<endl;
   //display data from stu (union)
   cout<<  "Display data from stu variable of union"<<endl;
   cout<< "stu.code= "<<stu.code<<endl;
   cout<< "stu.name= "<<stu.name<<endl;
   cout<< "stu.midterm= "<<stu.midterm<<endl;
   cout<< "stu.final= "<<stu.final<<endl;
   cout<< "stu.grade= "<<stu.grade<<endl;
   cout<< "*** error value of stu ***\a"<<endl;
   getch();
}
```

♦ แบบฝึกหัดท้ายบท

1. ให้เขียนโปรแกรมเพื่อเก็บประวัติพนักงานของบริษัทแห่งหนึ่งมีจำนวนพนักงานสูงสุดไม่เกิน 100 คน รายละเอียดข้อมูลที่จัดเก็บได้แก่ รหัสพนักงาน, ชื่อสกุล, ที่อยู่, อายุ, เงินเดือน,ตำแหน่ง, สถานภาพสมรส กำหนดการจัดเก็บรายละเอียดข้อมูลเป็นชนิดโครงสร้าง โดยสร้างเป็นฟังก์ชันต่าง ๆ ให้ทำงาน ดังนี้
   - กรอกข้อมูลได้ตามจำนวนคนที่ต้องการ (แต่ไม่เกิน 100 คน)
   - แสดงรายงานรายละเอียดประวัติของพนักงานทุกคนได้, แสดงอายุเฉลี่ย, เงินเดือนรวม, เงินเดือนเฉลี่ย และคำนวณภาษีหัก ณ ที่จ่าย 5% ของเงินเดือนพนักงานแต่ละคน
   - ค้นหาประวัติพนักงานเป็นรายบุคล โดยการกรอกรหัสพนักงานเพื่อค้นหาได้

2. ให้นักศึกษาเขียนโปรแกรมเพื่อคำนวณการตัดเกรดนักศึกษา โดยจัดเก็บข้อมูลเป็นชนิดโครงสร้าง จำนวนไม่เกิน 50 คน

   กำหนดเงื่อนไขการตัดเกรด ดังนี้

คะแนนระหว่าง	1-49	เกรด	F
คะแนนระหว่าง	50-59	เกรด	D
คะแนนระหว่าง	60-69	เกรด	C
คะแนนระหว่าง	70-79	เกรด	B
คะแนนระหว่าง	80-100	เกรด	A

   โปรแกรมมีความสามารถดังนี้

- กรอกรายละเอียด รหัส, ชื่อนักศึกษา, คะแนนระหว่างภาค, คะแนนปลายภาค โดยสามารถกำหนดจำนวนคนที่ต้องการกรอกข้อมูลได้ (แต่ไม่เกิน 50 คน) หรือสามารถหยุดกรอกข้อมูลได้โดยที่ไม่จำเป็นต้องกรอกครบ 50 คน
- รวมคะแนนเพื่อนำไปคำนวณตัดเกรด
- คำนวณการตัดเกรดตามเกณฑ์ที่กำหนดไว้
- นับจำนวนคนที่ได้รับเกรดแต่ละเกรด
- แสดงผลการตัดเกรดเรียงตามคะแนนจากมากไปหาน้อย

โดยที่การเขียนโปรแกรมจะต้องสร้างเป็นฟังก์ชันแบ่งตามหน้าที่การทำงานในโปรแกรมให้เหมาะสม