บทที่ 11
ระเบียน
(Records)

ในบทนี้จะแนะนำแบบชนิดข้อมูลเชิงโครงสร้างชนิดใหม่ ได้แก่ ระเบียนซึ่งคล้ายกับแถวลำดับ ระเบียนประกอบด้วย หน่วยข้อมูลที่สัมพันธ์กัน แต่สิ่งที่ไม่เหมือนแถวลำดับคือส่วนประกอบของระเบียนอาจเป็นข้อมูลชนิดแตกต่างกันได้ เราใช้ระเบียนเพื่อเก็บหน่วยที่หลากหลายของสารสนเทศเกี่ยวกับคน ตัวอย่างเช่น ชื่อ เลขที่อยู่ ปีที่เกิด และเพศ

เราไม่สามารถเก็บสารสนเทศนี้ในแถวลำดับ เพราะว่า ชื่อ และเลขที่อยู่ เป็นสายอักขระ ปีที่เกิดเป็นเลขจำนวนเต็ม และเพศเป็นตัวอักขระ

สุดท้าย เราสามารถใช้แถวลำดับซึ่งมีสมาชิกเป็นระเบียน และระเบียนซึ่งมีส่วนประกอบเป็นระเบียนอื่นๆ เพื่อแทนองค์กรที่ซับซ้อนมากกว่าของสารสนเทศ

11.1 แบบชนิดข้อมูลระเบียน (The Record Data Type)

ถึงแม้ว่าแถวลำดับจะเป็นโครงสร้างข้อมูลที่มีประโยชน์ ซึ่งสามารถรวบรวมข้อมูลที่เกี่ยวข้องกัน แต่มีข้อจำกัดเพราะว่าข้อมูลทั้งหมดที่เก็บต้องเป็นชนิดเดียวกัน ปัญหาการเขียนโปรแกรมจำนวนมากสามารถแก้ไขได้ง่าย ถ้าเราสามารถจัดการข้อมูลซึ่งเกี่ยวข้องกันให้มีแบบชนิดข้อมูลแตกต่างกัน ด้วยโครงสร้างข้อมูล ระเบียน เราสามารถเก็บข้อมูลที่เกี่ยว-ข้องกัน ซึ่งมีชนิดแตกต่างกันในหนึ่งโครงสร้างข้อมูล ในหัวข้อนี้จะแสดงให้เห็นว่าจะประกาศและประมวลผลระเบียนอย่างไร

ระเบียน หมายถึง กลุ่มของหน่วยข้อมูลที่เกี่ยวข้องกัน มีชนิดแตกต่างกัน (Record is an aggregate of related data items of different types.)

การประกาศชนิดระเบียน (Declaring Record Type)

เพื่อให้เป็นระเบียบ การเข้าถึงสารสนเทศสมาชิกผู้ร่วมงาน ผู้จัดการแผนกซอฟต์แวร์ต้องการให้เก็บสารสนเทศของพนักงานแต่ละคนดังนี้

ID : 1234

Name : Caryn Jackson

Gender : F

Number of Dependents : 2

Hourly Rate : 6.00

Total Wage : 240.00

เราสามารถเก็บสารสนเทศนี้ในหนึ่งระเบียนที่มีสมาชิกข้อมูลหกตัว แบบชนิดข้อมูลระเบียนหมายถึงแผ่นแบบซึ่งอธิบายรูปแบบของแต่ละระเบียน ได้แก่ ชื่อและชนิดของสมาชิกข้อมูลแต่ละตัว หรือเขตข้อมูล (field) ของระเบียน การเก็บสารสนเทศข้างต้นนี้ เราใช้ระเบียนชนิด Employee ซึ่งแสดงให้เห็นถัดไป มีแต่ข้อมูลจำนวนเต็มสองชุด เขตข้อมูลสายอักขระ เขตข้อมูลตัวอักขระ (สำหรับเพศ) และเขตข้อมูลจำนวนจริงสองชุด

```
const
   StringLength  =  20;

type
   IDRange       =  1111 .. 9999;
   StringType    =  string [stringLength];

   Employee      =  record
                       ID     :  IDRange;
                       Name   :  StringType;
                       Gender :  Char;
                       NumDepend :  Integer;
```

```
    Rate, TotWages : Real
end; {Employee}
```

การประกาศชนิดสำหรับระเบียน Employee ให้ปิดล้อมด้วยคำว่า record และ end คำอธิบายเขตข้อมูลแต่ละตัวประกอบด้วย ชื่อเขตข้อมูล (เป็นไอเดนติไฟเออร์ของ Pascal) ตามด้วยแบบชนิดข้อมูลของเขตข้อมูล เครื่องหมาย colon (:) ใช้คั่นชื่อเขตข้อมูลออกจากแบบชนิดข้อมูล

แบบชนิดข้อมูลของเขตข้อมูลอาจเป็นแบบชนิดข้อมูลมาตรฐาน (Char, Integer, Real) ชนิดอย่างง่ายให้นิยามโดยผู้ใช้ (IDRange) หรือแบบชนิดโครงสร้าง (StringType)

การประกาศตัวแปรใช้สำหรับจัดสรรเนื้อที่หน่วยเก็บให้กับระเบียน ตัวแปร ระเบียน Clerk (ดูรูป 11.1) ประกาศดังนี้

```
var
   Clerk : Employee;
```

ตัวแปรระเบียน Clerk มีโครงสร้างซึ่งระบุในการประกาศสำหรับชนิดระเบียน Employee ดังนั้น หน่วยความจำซึ่งถูกจัดสรรประกอบด้วยพื้นที่หน่วยเก็บสำหรับค่าหกค่าที่แตกต่างกัน

สำหรับไอเดนติไฟเออร์ตัวอื่นๆ ในโปรแกรม Pascal ชื่อเขตข้อมูลควรจะพรรณนาสารสนเทศที่เก็บและชนิดข้อมูลที่ถูกต้องสำหรับชนิดของสารสนเทศนั้น ตัวอย่างเช่น ชนิดระเบียน Employee ใช้เขตข้อมูลสายอักขระ สำหรับเก็บชื่อของพนักงาน

แบบชนิดข้อมูลระเบียน หมายถึง แผ่นแบบซึ่งอธิบายรูปแบบของระเบียน ได้แก่ ชื่อและชนิดของสมาชิกข้อมูลแต่ละตัว (Record data type is a template that describes the format of a record including the name and type of each data element.)

เขตข้อมูลระเบียน หมายถึง สมาชิกข้อมูลในระเบียน (Record field is a data element in a record.)

Syntax Display

การประกาศชนิดระเบียน (Record Type Declaration)

Form :

```
type
    rectype = record
                  idlist1 : type1;
                  idlist2 : type2;
                     . . .
                  idlistn : typen
              end;
```

ตัวอย่าง

```
type
    Complex  = record
                   RealPart, ImaginaryPart : Real
               end;
```

มีความหมายดังนี้ : ไอเดนติไฟเออร์ rectype เป็นชื่อของโครงสร้างระเบียนซึ่งกำลังจะอธิบาย idlisht แต่ละตัวคือรายการของชื่อเขตข้อมูลหนึ่งชื่อหรือมากกว่าหนึ่งชื่อ แต่ละตัวคั่นด้วยเครื่องหมาย comma แบบชนิดของเขตข้อมูลแต่ละตัว ใน idlist กำหนดโดย type

ข้อสังเกต ชื่อเขตข้อมูลต้องเป็นเพียงหนึ่งเดียว (unique) ภายในชนิดระเบียน type แต่ละตัวอาจเป็นแบบชนิดข้อมูลมาตรฐาน หรือชนิดให้นิยามโดยผู้ใช้ อาจถูกนิยามก่อนระเบียนหรืออาจเป็นส่วนหนึ่งของคำอธิบายระเบียน

Clerk. ID	1234
Clerk. Name	Caryn Jackson
Clerk. Gender	F
Clerk. NumDepend	2
Clerk. Rate	6.00
Clerk. Totwages	240.00

รูป 11.1 ตัวแปรระเบียน Clerk

การเข้าถึงเขตข้อมูลระเบียน (Accessing Record Fields)

เราสามารถเข้าถึงเขตข้อมูลระเบียน โดยการใช้ ตัวเลือกเขตข้อมูล (field selector) ซึ่งประกอบด้วย ชื่อตัวแปรระเบียน ตามด้วย ชื่อเขตข้อมูล ใส่จุด (period) คั่นระหว่างชื่อระเบียนกับชื่อเขตข้อมูล

```
record – variable. field – name
```

ตัวอย่าง 11.1

ข้อความสั่งต่อไปนี้เก็บข้อมูลในตัวแปรระเบียน Clerk รูป 11.1 แสดง content ของ Clerk หลังจากข้อความสั่งเหล่านี้ถูกกระทำการ

```
Clerk. ID          := 1234 ;          {Store 1234 in ID field}
Clerk. Name        := 'Caryn Jackson; {Store string in name field}
Clerk. Gender      := 'F';            {Store F in Gender field}
Clerk. NumDepend   := 2;              {Store 2 in NumDepent field}
Clerk. Rate        := 6.00;           {Store 6.00 in Rate field}
Clerk. TotWages    := 0.0;            {Initialize TotWage field}
Clerk. TotWages    := Clerk. TotWages + 40.0 * Clerk. Rate
```

เมื่อข้อมูลถูกเก็บในระเบียนแล้ว มันจะถูกจัดดำเนินการในวิธีเดียวกับข้อมูลอื่นๆ ในหน่วยความจำ ตัวอย่างเช่น ข้อความสั่งกำหนดค่าสุดท้ายในตัวอย่างข้างต้น คำนวณค่าจ้างทั้งหมดตัวใหม่ของ Clerk โดยการบวกค่าจ้างของสัปดาห์นี้กับค่าจ้างทั้งหมดก่อนหน้านั้น (ในกรณีนี้คือ 0.0) ผลลัพธ์จากการคำนวณเก็บในเขตข้อมูลระเบียน Clerk.TotWages

ข้อความสั่ง

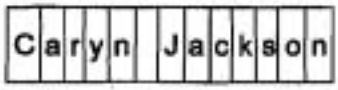
```
Write ('The Clerk is');
case Clerk. Gender of
   'F', 'f'   :  Write ('Ms.');
   'M', 'm'   :  Write ('Mr.')
end;  {case}
Write (Clerk. Name)
```

แสดงชื่อของ clerk หลังจากมีคำนำหน้าถูกต้อง ('Ms.' หรือ 'Mr.') บรรทัดเอาต์พุตเป็นดังนี้

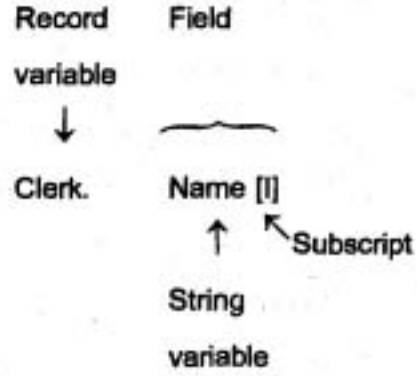
The clerk is Ms. Caryn Jackson

แถวลำดับเป็นเขตข้อมูลระเบียน (Arrays as Record Fields)

เขตข้อมูล Name ของระเบียน Clerk เป็นสายอักขระ ซึ่งที่จริงแล้วคือแถวลำดับของตัวอักขระ



เราสามารถอ้างถึงอักขระแต่ละตัวในเขตข้อมูลสายอักขระ (หรือเขตข้อมูลแถวลำดับ) โดยการเขียนตัวเลือกเขตข้อมูล ตามด้วยตรรชนีล่างอยู่ภายในวงเล็บก้ามปู (brackets) ดังนี้



ฟังก์ชัน designator Length (Clerk. Name) กลับคืนจำนวนของตัวอักขระซึ่งเก็บในเขตข้อมูล Clerk. Name for ลูป ซึ่งเข้าถึงอักขระแต่ละตัว จากอักขระตัวแรก (C) ไปจนถึงอักขระตัวสุดท้าย (n) :

```
Write ('The clerk' 's initials are : ');
for I := 1 to Length (Clerk. Name) do
   if  (Clerk. Name [I] >= 'A') and
       (Clerk. Name [I] <= 'Z' then
       Write (Clerk. Name [I]);
WriteLn
```

เงื่อนไขของข้อความสั่ง if เป็นจริง ถ้า Clerk. Name [I] เป็นอักษรตัวพิมพ์ใหญ่ (uppercase letter) ดังนั้น ส่วนของโปรแกรมแสดงบรรทัดเอาต์พุตดังนี้

The clerk's initials anr : CJ

แบบฝึกหัด 11.1 Self-Check

1. แต่ละส่วนในสินค้าคงคลัง (inventory) แห่งหนึ่ง แทนด้วยหมายเลขสินค้า (part number) ชื่อสินค้า (descriptive name) ปริมาณสินค้าคงเหลือ (quantity on hand) และราคา จงให้นิยามชนิดระเบียน ชื่อ Part

2. โปรแกรมกราฟิกคอมพิวเตอร์ซึ่งคำนวณความเข้ม (intensity) ของจุดภาพ (pixels) บนจอภาพ จงให้นิยามชนิดระเบียนชื่อ Pixels สำหรับโปรแกรมนี้ ซึ่งจะใช้แทนจุดภาพ เมื่อจุดภาพแต่ละจุดมีตำแหน่งเลขจำนวนเต็ม X และ Y และค่าความเข้มจำนวนจริงสำหรับสี แดง, เขียว และน้ำเงิน

11.2 ระเบียนเป็นตัวถูกดำเนินการและพารามิเตอร์ (Records as Operands and Parameters)

นอกจากการเข้าถึงเขตข้อมูลระเบียนแต่ละตัวเพื่อกระทำการดำเนินการคำนวณ การดำเนินการตรรกะ หรือการดำเนินการอินพุต หรือเอาต์พุตแล้ว บ่อยครั้งที่โปรแกรมเมอร์จำเป็นต้องเขียนกระบวนงานสั่งซึ่งประมวลผลทั้งระเบียน (entire records) โดยปกติจะสะดวกมากกว่าในการส่งระเบียนตัวมันเองเป็นพารามิเตอร์ของกระบวนงาน ไม่ใช่ส่งเขตข้อมูลแต่ละตัว โปรแกรมเมอร์จำเป็นต้องทำสำเนา (copy) content ของตัวแปรระเบียนหนึ่งตัว ไปยังตัวแปรระเบียนอีกหนึ่งตัว ในหัวข้อนี้จะอธิบายการดำเนินการบนทั้งระเบียน

การทำสำเนาหรือการกำหนดค่าระเบียน (Record Copy or Assignment)

เราสามารถทำสำเนาเขตข้อมูลทั้งหมดของหนึ่งตัวแปรระเบียน ไปยังตัวแปรระเบียนอีกหนึ่งตัวที่มีชนิดเดียวกันได้ ด้วยข้อความสั่ง (กำหนดค่า) ทำสำเนาระเบียน

ตัวอย่างเช่น ถ้า Clerk และ Janitor ทั้งคู่เป็นตัวแปรระเบียนชนิด Employee ข้อความสั่ง

```
Clerk := Janitor   {copy Janitor to Clerk}
```

ทำสำเนาเขตข้อมูลแต่ละตัวของ Janitor ไปยังเขตข้อมูลที่สมนัยกันของ Clerk

ระเบียบเป็นพารามิเตอร์ (Records as Parameters)

ระเบียนสามารถถูกสั่งเป็นพารามิเตอร์ไปยังฟังก์ชันหรือกระบวนงานจัดหาพารามิเตอร์จริง เป็นชนิดเหมือนกับพารามิเตอร์รูปนัยซึ่งสมนัยของมัน การใช้ระเบียนเป็นพารามิเตอร์ทำให้รายการพารามิเตอร์สั้นลง เพราะว่า พารามิเตอร์หนึ่งตัว (ตัวแปรระเบียน) สามารถส่งพารามิเตอร์ที่เกี่ยวข้องกันได้หลายตัว

เราไม่สามารถใช้กระบวนงานอินพุตของ Pascal เพื่ออ่านข้อมูลจากคีย์บอรด หรือแฟ้มข้อความไปทั้งหนึ่งระเบียน (an entire record) ในทำนองเดียวกัน เราไม่สามารถใช้กระบวนงานเอาต์พุตของ Pascal เพื่อเขียน (write) content ของทั้งหนึ่งระเบียนไปยังจอภาพหรือแฟ้มข้อความ เราจำเป็นจะต้องลงรหัสกระบวนงานของเราเองเพื่อกระทำการดำเนินการร่วมเหล่านี้ เช่นที่แสดงให้เห็นในรูป 11.2 และ 11.3

แต่ละกระบวนงานมีพารามิเตอร์หนึ่งตัว : ได้แก่ ระเบียนซึ่งกำลังจะถูกอ่าน หรือเขียน

ตัวอย่าง 11.2 การอ่านหนึ่งระเบียน (Reading a Record) กระบวนงาน Read Employee ในรูป 11.2 อ่านข้อมูลไปยังเขตข้อมูลทั้งหมดของหนึ่งตัวแปรระเบียน ชนิด Employee ข้อความสั่งเรียกกระบวนงาน

ReadEmployee (Clerk)

ทำให้ข้อมูลซึ่งอ่านเข้ามาเก็บในตัวแปรระเบียนชื่อ Clerk

```
precedure ReadEmployee (var AnEmp {output} : Employee);
{
   Reads one employee record into AnEmp.
   Pre  :  None.
   Post :  Data are read into record AnEmp.
}
begin {ReadEmployee}
   Write ('ID> ');
   ReadLn (AnEmp. ID);
   Write ('Name> ');
   ReadLn (AnEmp. Name);
   Write ('Gender (F or M) > ');
   ReadLn (AnEmp. Gender);
   Write ('Number of dependents> ');
   ReadLn (AnEmp. NumDepend);
```

```
    Write ('Hourly rate > ');
    ReadLn (AnEmp. Rate);
    Write ('Total wage to date > ');
    ReadLn (AnEmp. TotWages)
end; {ReadEmployee}
```

รูป 11.2 กระบวนงาน ReadEmployee

ตัวอย่าง 11.3 การเขียนหนึ่งระเบียน (Writing a Record) ในโปรแกรมกำหนดเกรด ข้อสรุปสถิติ (statistics) สำหรับคะแนนสอบ ได้แก่ คะแนนเฉลี่ย คะแนนสูงสุด คะแนนต่ำสุด และส่วนเบี่ยงเบนมาตรฐาน ก่อนหน้านี้เราเก็บข้อมูลเหล่านี้ในตัวแปรแยกจากกัน แต่ขณะนี้เราสามารถจัดกลุ่ม (group) เข้าไว้ด้วยกันเป็นหนึ่งระเบียน ดังนี้

```
type
    ScoreRange   = 0 . . 100;
    ExamStats    = record
                       Low, High  :  ScoreRange;
                       Average, StandardDev : Real
                   end; {ExamStats}
```

กระบวนงาน PrintStat (ในรูป 11.3) แสดงผลค่าต่างๆ ซึ่งเก็บในเขตข้อมูลแต่ละตัวของพารามิเตอร์ระเบียนของมัน (ชนิด ExamStats)

```
procedure PrintStat (Exam {input} : ExamStats);
{
    Prints the exam statistics.
    Pre   :  The fields of record parameter Exam are assigned values.
    Post  :  Each field of Exam is displayed.
}
begin {PrintStat}
```

```
    WriteLn ('High core : ', Exam. High : 1);
    WriteLn ('Low score : ', Exam. Low : 1);
    WriteLn ('Average : ', Exam. Average : 3 : 1);
    WriteLn ('Standard deviation : ', Exam. StandardDev : 3 : 1)
end; {PrintStat}
```

รูป 11.3 กระบวนงาน PrintStat

แบบฝึกหัด 11.2 Self-Check

1. ส่วนของโปรแกรมข้างล่างนี้ทำงานอะไรจงจัดหาการประกาศสำหรับตัวแปร Exam1 และ Exam2

```
PrintStat (Exam1);
Exam2 := Exam1;
Exam2. High := Exam2. High - 5.0;
PrintStat (Exam2)
```

2. จงอธิบายว่าแต่ละข้อย่อยต่อไปนี้ชุดใดไม่ถูกต้อง สำหรับ Exam1 และ Exam2 จากแบบฝึกหัดข้อ 1 จากนั้นให้แก้ไขข้อความสั่งชุดซึ่งไม่ถูกต้องให้ถูกต้อง

a) var Exam1, Exam2 : Exam;

b) Exam1 := Exam2

c) ReadLn (Exam1. Average)

d) Exam1. High := Exam2

e) Exam2. Low := Exam1. High

f) Exam2. Low := Exam1. Average

g) WriteLn (Exam1)

เขียนโปรแกรม

1. จงเขียนกระบวนงานซึ่งอ่านข้อมูลสำหรับจุดภาพ (pixel) และกลับคืนข้อมูลผ่านพารามิเตอร์ของมัน (ดู Self-Check ข้อ 2 สำหรับหัวข้อ 11.1)

11.3 ข้อความสั่ง with (The with Statement)

การเขียนตัวเลือกเขตข้อมูลที่สมบูรณ์ (complete field selector) แต่ละครั้งเราต้องอ้างถึงเขตข้อมูลของระเบียนซึ่งเป็นงานที่น่าเบื่อหน่าย เราสามารถใช้ข้อความสั่ง with เพื่อทำให้ตัวเลือกเขตข้อมูลสั้นลง ข้อความสั่ง with ประมวลผลตัวแปรระเบียน Clerk ที่แสดงในรูป 11:1 ดังนี้

```
with Clerk do
   begin
      Write ('The clerk is');
      case Gender of
         'F', 'f' : Write ('Ms.');
         'M', 'm' : Write ('Mr.');
      end; {case}
      WriteLn (Name);

      TotWages := TotWages + 40.0 * Rate;
      WriteLn ('The clerk' 's total wages are $', TotWages : 4 : 2)
   end; {with}
```

จะเห็นว่า เราไม่จำเป็นต้องระบุทั้งตัวแปรระเบียนและชื่อเขตข้อมูลภายในข้อความสั่ง with เพราะว่า ตัวแปรระเบียน Clerk ซึ่งกำหนดไว้ในหัวเรื่องของข้อความสั่ง with จึงมีเฉพาะชื่อเขตข้อมูลเท่านั้นที่จำเป็น ไม่ใช่ตัวเลือกเขตข้อมูลที่สมบูรณ์ (ตัวอย่างเช่น Rate แทนที่จะเป็น Clerk. Rate) ข้อความสั่ง with มีประโยชน์โดยแฉพาะเมื่อเราจัดดำเนินการกับเขตข้อมูลหลายตัวของตัวแปรระเบียนชุดเดียวกัน

Syntax Display

ข้อความสั่ง with

Form :

```
with record var do
   statement
```

ตัวอย่าง

```
with Clerk do
    if NumDepend > 3 then
        Rate := 1.5 * Rate
```

มีความหมายดังนี้ : statement อาจจะเป็น single statement หรือ compound statement

record var เป็นชื่อของตัวแปรระเบียน

ที่ใดก็ตามภายใน statement เราสามารถอ้างถึงเขตข้อมูลของ record var ได้โดยกำหนดเฉพาะชื่อเขตข้อมูลเท่านั้น

ตัวอย่าง 11.4

โปรแกรมจำนวนมากจำเป็นต้องเก็บ track ของเวลาของวัน (the time of day) และปกติทำสิ่งนี้โดยการปรับเวลาของวัน หลังจากหนึ่งช่วงเวลาผ่านไป ระเบียนชนิด Time ซึ่งประกาศข้างล่างนี้ สมมติว่า

นาฬิกา 24 ชั่วโมง

```
type
    Time =  record
                Hour :  0 . . 23;
                Minute : 0 . . 59
            end; {Time}
```

กระบวนงาน ChangeTime ในรูป 11.4 ปรับเป็นปัจจุบัน (updates) เวลาของวัน TimeOfDay (ชนิด Time) หลังจากหนึ่งช่วงเวลา ElapsedTime ซึ่งแสดงในหน่วยวินาที (seconds)

ข้อความสั่งแต่ละชุดใช้ตัวดำเนินการ mod ปรับปัจจุบันเขตข้อมูลเฉพาะของระเบียน ซึ่งแทนด้วย TimeOfDay

ตัวดำเนินการ mod ทำให้มั่นใจว่า ค่าปรับปัจจุบันแต่ละค่าอยู่ภายในพิสัยที่ต้องการ

ตัวดำเนินการ div เปลี่ยนพหุคูณ (multiples) ของ 60 วินาที ให้เป็นนาทีและพหุคูณของ 60 นาที ให้เป็นชั่วโมง

การใช้ข้อความสั่ง with ทำให้การเขียนกระบวนงาน ChangeTime ง่ายขึ้น ถ้าไม่มีข้อความสั่ง with ข้อความสั่งกำหนดค่า บรรทัดแรกต้องเขียนดังนี้

```
NewSec := TimeOfDay.Second + ElapsedTime; {total records}
```

```
procedure ChangeTime (ElapsedTime {input} : Integer;
                      var TimeOfDay {input/output} : Time);
{
Updates the time of day, TimeOfDay, assuming a 24-hour clock and an
elapsed time of ElapseTime in seconds.
Pre  :  ElapsedTime and record TimeOfDay are assigned values.
Post :  TimeOfDay is incsemented by ElapsedTime.
}

var
   NewHour, NewMin, NewSec : Integer; {temporary variables}

begin {ChangeTime}
   with TimeOfDay do
      begin
         NewSec := Second + ElapsedTime; {total seconds}
         Second := NewSec mod 60; {seconds over 60}
         NewMin := Minute + (NewSec div 60); {total minutes}
         Minute := NewMin mod 60; {minutes over 60}
         NewHour := Hour + (NewMin div 60); {total hours}
         Hour := NewHour mod 24 {hours over 24}
      end {with}
end; {ChangeTime}
```

รูป 11.4 กระบวนงาน ChangeTime

สไตล์ของโปรแกรม (Program Style)

ข้อควรระวังเกี่ยวกับข้อความสั่ง with (A word of Caution About the with statement)

ถึงแม้ว่าข้อความสั่ง with มีประโยชน์ในการลดความยาวของข้อความสั่งโปรแกรม ซึ่งจัดดำเนินการส่วนประกอบของระเบียน แต่มันลดความชัดเจนของข้อความสั่งเหล่านั้นด้วยเช่นกัน ตัวอย่างเช่น ในข้อความสั่งกำหนดค่าก่อนหน้านี้ ตัวเลือกเขตข้อมูล TimeOf Day.Second แสดงอย่างชัดเจนว่า Second เป็นเขตข้อมูลของพารามิเตอร์ระเบียน TimeOf Day ซึ่งสิ่งนี้ไม่ได้เป็นเอกสารอย่างชัดเจนในข้อความสั่ง

NewSec := second + ElapsedTime; {total seconds}

ที่ใช้ในกระบวนงาน ChangeTime

แบบฝึกหัด 11.3 Self-Check

1. ถ้ากระบวนงานประกาศตัวแปรเฉพาะที่ (local variable) ชื่อ Second การใช้ข้อความสั่ง with จะมีข้อจำกัดอย่างไร ถ้าตัวแปรชนิด Time มีการประกาศด้วยเช่นกัน

2. ถ้าเขตข้อมูลทั้งหมดของตัวแปร New (ชนิด Time) กำหนดค่าเริ่มต้นเท่ากับ 0 การกระทำการของส่วนคำสั่งข้างล่างนี้ จะเปลี่ยนแปลง New อย่างไร

```
ChangeTime (3600, Now);
ChangeTime (7125, Now);
```

เขียนโปรแกรม

1. จงเขียนกระบวนงานซึ่งกำหนดค่าเริ่มต้นให้กับเขตข้อมูลทุกตัวของตัวแปร ชนิด Time เท่ากับ 0

2. จงเขียนฟังก์ชันซึ่งยอมรับพารามิเตอร์ชนิด Time และคำนวนเวลาที่สมมูลกันมีหน่วยเป็นวินาที กลับคืนผลลัพธ์เป็นชนิด LongInt (ให้อธิบาย)

11.4 แถวลำดับของระเบียน (Arrays of Records)

แถวลำดับและระเบียนเป็นโครงสร้างข้อมูลพื้นฐานใน Pascal คล้ายกับที่เราใช้ส่วนจำเพาะโปรแกรมพื้นฐาน กระบวนงานและฟังก์ชันเป็นบล็อกการสร้างของโปรแกรมที่มีขนาดใหญ่กว่า เราสามารถใช้โครงสร้างข้อมูลพื้นฐานเหล่านี้เป็นแบบจำลองการจัดระเบียบที่ซับซ้อนมากกว่าของข้อมูล

การเก็บฐานข้อมูลเป็นแถวลำดับของระเบียบ

วิธีร่วมอย่างหนึ่งของการจัดการข้อมูลในชีวิตจริงคือเพื่อสร้างฐานข้อมูล ซึ่งเป็นกลุ่มของข้อเท็จจริง หรือสารสนเทศเกี่ยวกับหน่วยข้อมูลที่เกี่ยวข้องกัน ตัวอย่างเช่น ในโรงพยาบาล การบำรุงรักษาฐานข้อมูลของพนักงาน ฐานข้อมูลของคนไข้ และฐานข้อมูลของยาที่สั่งจ่ายให้คนไข้ เราสามารถใช้ระเบียนเพื่อแทนหน่วยข้อมูลแต่ละตัวในฐานข้อมูล และเราสามารถใช้แถวลำดับของระเบียน เพื่อเก็บฐานข้อมูลแต่ละชุดในหน่วยความจำ ปอยครั้งที่เราอ่านฐานข้อมูลจากแฟ้มดิสก์ก่อนที่จะเข้าถึงมัน และเราเขียนฐานข้อมูลออกจากดิสก์เมื่อเราปรับให้มันเป็นปัจจุบันเสร็จสิ้นแล้ว

ฐานข้อมูล หมายถึง กลุ่มของข้อเท็จจริงหรือสารสนเทศเกี่ยวกับหน่วยข้อมูลที่เกี่ยวข้องกัน ปกติจัดการเป็นแฟ้มหรือตารางของระเบียน (Database is a collection of facts or information about related items usually organized as a file or table of records.)

ตัวอย่างอย่างง่ายของฐานข้อมูล ได้แก่ กลุ่มของระเบียนนักศึกษาสำหรับหนึ่งห้องเรียน สำหรับนักศึกษาแต่ละคน เราต้องการเก็บชื่อของนักศึกษา คะแนนสอบสามครั้ง คะแนนเฉลี่ย และเกรดของนักศึกษา เราสามารถเก็บสารสนเทศของนักศึกษาแต่ละคนในระเบียนชนิด Student โครงสร้างข้อมูลซึ่งประกาศถัดไปนี้เก็บข้อมูลห้องเรียนในแถวลำดับของระเบียนนักศึกษา แถวลำดับ Class แสดงในรูป 11.5

```
const
   MaxClass = 200;
   StringLength = 20;
   NumberExams = 3;

type
   ClassIndex =  1..MaxClass;
   StringType =  string [StringLength];
   ScoreArray =  array [1..NumberExams] of Integer;
   Student    =  record
                    Name    :  StringType;
                    Scores  :  ScoreArray;
                    Average :  Real;
                    Grade   :  Char
```

```
        end; {Student}
    StudentArray = array [ClassIndex] of Student;

var
    Class : StudentArray;
```

ในรูป 11.5 ข้อมูลของนักศึกษาคนแรกเก็บในระเบียน Class[1] หน่วยข้อมูลแต่ละตัว ได้แก่ Class[1].Name.

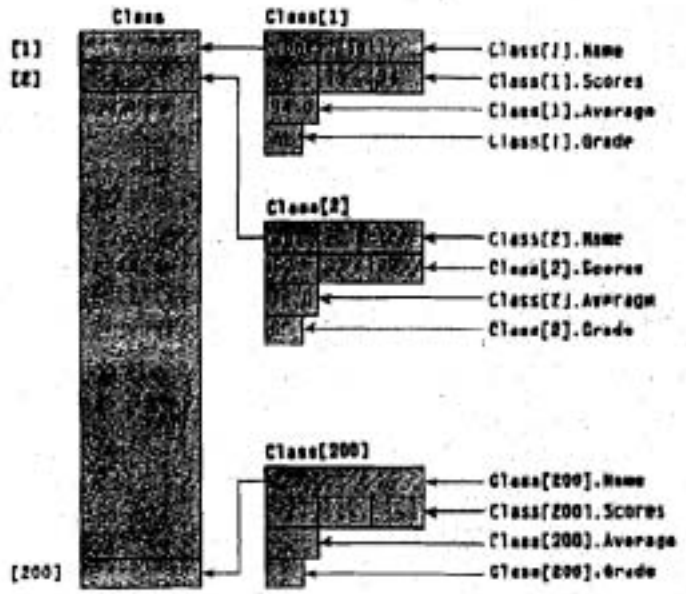
Class[1]. Score[1], Class[1]. Score[2],

Class[1]. Score[3], Class[1]. Average และ

Class[1]. Grade โปรดสังเกตว่า จำเป็นต้องใช้ตรรชนีล่างสองตัวเพื่อเข้าถึงคะแนนสอบ ตรรชนีล่างตัวแรกเลือกสมาชิกของแถวลำดับ Class ส่วนตรรชนีล่างตัวที่สองเลือกสมาชิกของแถวลำดับ Score ตัวอย่างเช่น Class[1]. Scores[1] คือ 90

ตัวอย่าง 11.5

กระบวนงาน ReadClas (รูป 11.6) อ่านข้อมูลไว้ในแถวลำดับของระเบียนนักศึกษา กระบวนงานนี้กลับคืนผลลัพธ์สองค่า ได้แก่ แถวลำดับของข้อมูลนักศึกษาและจำนวนของระเบียนนักศึกษาซึ่งถูกเก็บไว้



รูป 11.5 แถวลำดับของระเบียน

ข้อความสั่ง for

```
for I := 1 to ClassSize do
    ReadStudent (Class[I])
```

ใส่ข้อมูลในแถวลำดับ Class ทุกครั้งที่ ReadStudent (ดูแบบฝึกหัดเขียนโปรแกรมข้อ 1 ตอนท้ายของหัวข้อนี้) ถูกเรียก, ระเบียนกลับคืน (ชนิด Student) จะถูกเก็บเป็นสมาชิกตัวที่ I (1 ≤ I ≤ ClassSize) ของแถวลำดับ Class รายละเอียดของการอ่านแต่ละระเบียนทิ้งไว้ในกระบวนงาน ReadStudent

---

```
procedure ReadClass  (var Class {output} : StudentArray;
                      var ClassSize {output} : Integer);
{  Returns the number of students in the class through ClassSize. Fills the
   subarray Class [1..ClassSize] with student data.
   Pre   :  None
   Post  :  1 <= ClassSize <= MaxClass and Class[I] contains the input data
            for the Ith student.
   Calls :  EnterInt (fig.9.6) and ReadStudent
}
   var
      I  :  Integer; {loop control and array subscript}

begin {ReadClass}
   WriteLn ('Enter number of students : ');
   EnterInt (1, MaxClass, ClassSize);

   {  Read the student data. }
   WriteLn ('Enter each student' 's data : ');
   for I := 1 :o ClassSize do
```

```
        ReadStudent (Class[I])
    end; {ReadClass}
```

รูป 11.6 กระบวนงาน ReadClass

การใช้ข้อความสั่ง with กับแถวลำดับของระเบียน (Using the with Statement with an Array of Records)

จงรอบคอบเมื่อใช้ข้อความสั่ง with เพื่อประมวลผลแถวลำดับของระเบียน ตัวอย่างเช่น ข้อความสั่ง with ซึ่งเริ่มต้น

```
with Class[I] do
```

ใช้ตัวแปรดรรชนีล่าง Class[I] เป็นตัวแปรระเบียนของมัน การอ้างถึงสมาชิกแถวลำดับโดยเฉพาะขึ้นอยู่กับค่าของ I ถ้า I ไม่ถูกนิยาม (undefined) หรือ I อยู่นอกพิสัย ผลลัพธ์จะเป็นข้อผิดพลาดเวลาดำเนินงาน (run-time error)

ถ้า I มีการ update ภายในข้อความสั่ง with การอ้างถึงสมาชิกแถวลำดับจะไม่เปลี่ยนแปลง ตัวอย่างเช่น ข้อความสั่งข้างล่างนี้แสดงผล ชื่อของนักศึกษาคนแรก จำนวน MaxClass ครั้ง เพราะว่า I เท่ากับ 1 เมื่อมาถึงข้อความสั่ง with

Class[1] จึงเป็นระเบียนที่ถูกอ้างถึงใน body ของข้อความสั่ง with ถึงแม้ว่า for ลูปจะมีการเปลี่ยนแปลงค่าของ I ก็ตาม Class[1] ยังคงเป็นระเบียนที่ถูกอ้างถึง ดังนั้น

Class[1]. Name จะถูกแสดงผลซ้ำๆ กัน

```
I := 1;
{incorrect attampt to display all student names}
with Class[I] do
   for I := 1 to ClassSize do
      WriteLn (Name)
```

วิธีที่ถูกต้องเพื่อจัดลำดับข้อความสั่งเหล่านี้ คือ

```
{Display all student names.}
for I := 1 to ClassSize do
   with Class [I] do
      WriteLn (Name)
```

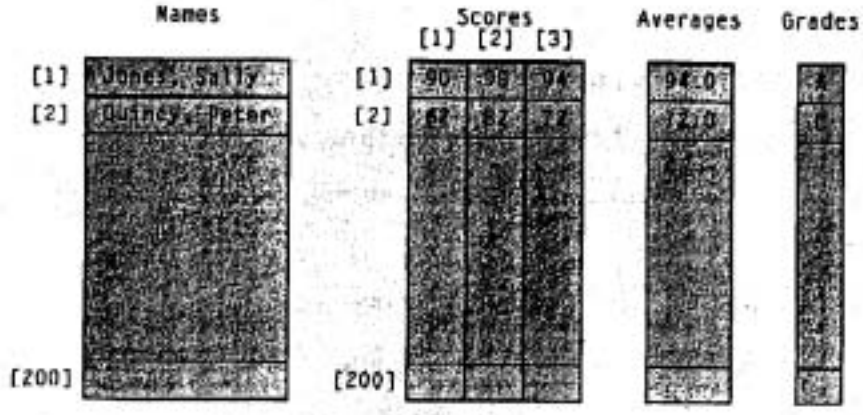
ขณะนี้ชื่อของนักศึกษาทั้งหมดจะถูกแสดงผล เพราะว่า I ถูกเปลี่ยนแปลงโดยข้อความสั่ง for ภายนอกข้อความสั่ง with แต่ละครั้งที่มาถึงข้อความสั่ง with มันอ้างถึงระเบียนชุดใหม่ เมื่อใดก็ตามที่ข้อความสั่ง for เข้าถึงแถวลำดับของระเบียนแบบเรียงอันดับ ข้อความสั่ง with ควรจะซ้อนใน ภายในข้อความสั่ง for

ผลกระทบของโครงสร้างข้อมูลบนการออกแบบโปรแกรม (Effect of Data Structures on Program Design)

-การเลือกโครงสร้างข้อมูลสำหรับปัญหา มีผลกระทบอย่างสำคัญบนการออกแบบโปรแกรม เมื่อเราลงรหัสกระบวนงาน ซึ่งประมวลผลข้อมูลห้องเรียน ข้อดีคือ ข้อมูลของนักศึกษาแต่ละคนเก็บในระเบียนแยกจากกัน และใช้กระบวนงานซึ่งเกี่ยวข้องกับชนิดระเบียน ตัวอย่างเช่น กระบวนงาน ReadClass เรียกกระบวนงาน ReadStudent เพื่ออ่านระเบียนนักศึกษาแต่ละคน บ่อยครั้งเราจะพบว่ากระบวนงานคล้ายกับ ReadStudent จะมีการลงรหัสเรียบร้อยแล้ว การเลือกโครงสร้างข้อมูลนี้ทำให้การลงรหัส ReadClass ง่ายขึ้น

อีกทางเลือกหนึ่งสำหรับการเก็บข้อมูลนักศึกษาคือจะใช้แถวลำดับสี่ชุดแยกจากกันดังนี้ แถวลำดับชุดที่หนึ่งมีสมาชิก MaxClass ตัวซึ่งเก็บชื่อของนักศึกษาทั้งหมด

ชุดที่สอง เป็นแถวลำดับสองมิติ ขนาด MaxClass x 3 เก็บคะแนนสอบ และแถวลำดับอีกสองชุด ที่มีสมาชิก MaxClass ตัวเก็บคะแนนเฉลี่ยและเกรดของนักศึกษาแต่ละคน ข้อมูลของนักศึกษาคนที่ I จะเก็บในแถวที่ I ของแถวลำดับทั้งสี่ชุด (ดูรูป 11.7)



รูป 11.7 แถวลำดับสี่ชุด สำหรับข้อมูลนักศึกษา

แบบฝึกหัด 1.4 Self-Check

1. สำหรับแถวลำดับของระเบียน Class (ดูรูป 11.5)

จงบอกค่าซึ่งจะถูกแสดงผลโดยข้อความสั่งที่ถูกต้องแต่ละคำสั่ง และแก้ไขข้อความสั่งที่ผิด (invalid) ให้ถูกต้อง

a) WriteLn (Class[1]. Name[4])

b) WriteLn (Class[1]. Score[4])

c) WriteLn (Class. Grade[3])

d) WriteLn (Class[1]. Name)

e) WriteLn (Class[1]. Name[4]. Grade)

f) WriteLn (Class[1]. Grade)

g) WriteLn (Class[1])

h) WriteLn (Class. Name[4])

11.5 ข้อผิดพลาดร่วมของการเขียนโปรแกรม (Common Programming Errors)

ข้อผิดพลาดร่วมส่วนใหญ่เมื่อใช้ระเบียนคือการกำหนดเขตข้อมูลของระเบียนที่จะให้จัดดำเนินการไม่ถูกต้อง

ตัวเลือกเขตข้อมูลแบบบริบูรณ์ (ตัวแปรระเบียนและชื่อเขตข้อมูลต้องนำมาใช้ยกเว้น การอ้างระเบียนซ้อนในอยู่ภายในข้อความสั่ง with หรือ ระเบียนทั้งหมดถูกจัดดำเนินการ

ถ้าชื่อตัวแปรระเบียน แสดงรายการในหัวเรื่องของข้อความสั่ง with, เฉพาะชื่อเขตข้อมูลเท่านั้น ซึ่งต้องใช้เพื่ออ้างถึงเขตข้อมูลของระเบียนนั้น ภายในข้อความสั่ง with เรายังคงต้องใช้ตัวเลือกเขตข้อมูลแบบบริบูรณ์ (full field selector) เพื่ออ้างถึงเขตข้อมูลของตัวแปรระเบียนอื่นๆ

ข้อผิดพลาดในแถวลำดับของระเบียน (Errors in Arrays of Records)

เมื่อแถวลำดับของระเบียนถูกประมวลผล ชื่อแถวลำดับ และดรรชนีล่าง ต้องมีอยู่เป็นส่วนของตัวเลือกเขตข้อมูล (ตัวอย่างเช่น X[I]. Key อ้างถึงเขตข้อมูล Key ของระเบียนที่ I)

ถ้าเราใช้ข้อความสั่ง for เพื่อประมวลผลสมาชิกแถวลำดับทั้งหมดตามลำดับ เราต้องซ้อนในข้อความสั่ง with ซึ่งอ้างถึงระเบียนแถวลำดับภายในข้อความสั่ง for เช่นที่แสดงข้างล่างนี้

```
for I := 1 to N do
   with X[I] do
```

ขณะที่ตัวแปรควบคุมลูป I เปลี่ยน, ระเบียนแถวลำดับตัวถัดไปจะถูกประมวลผลโดยข้อความสั่ง with

ถ้าอันดับการซ้อนในย้อนลำดับเป็นดังนี้

```
with X[I] do
   for I := 1 to N do
```

ระเบียนแถวลำดับชุดเดียวกันถูกประมวลผล N ครั้ง

ระเบียนที่ถูกประมวลผลกำหนดโดยค่าของ I เมื่อมาถึงข้อความ with ครั้งแรก การเปลี่ยนแปลงค่าของ I ภายในข้อความสั่ง with จะไม่มีผลกระทบ

ข้อสรุปของตัวสร้าง Pascal ตัวใหม่ (Summary of New Pascal Constructs)

Construct	Effect
การประกาศระเบียน	
type Part = record ID : 1111..9999; Quantity : Integer; Price : Real end; {Part}	ระเบียนชนิด Part ถูกประกาศด้วยเขตข้อมูลซึ่งสามารถเก็บจำนวนเต็มสองเขตข้อมูลและเลขจำนวนจริงหนึ่งเขตข้อมูล
var Nuts, Botts : Part;	Nuts และ Botts เป็นตัวแปรระเบียนชนิด Part
Record Reference	
TotalCost := Nuts.Quantity * Nuts. Price;	เขตข้อมูลสองชุดของ Nuts คูณกัน

Construct	Effect
WriteLn (Bolts. ID : 4)	พิมพ์เขตข้อมูล ID ของ Bolts
Record Copy	
Bolts := Nuts	ทำสำเนาระเบียน Nuts ไปที่ Bolts
ข้อความสั่ง with	
with Bolts do	
Write (ID : 4, Price : 8 : 2)	พิมพ์เขตข้อมูลสองเขตของ Bolts
การประกาศแถวลำดับของระเบียน	
type	
AElement = record	DataArray เป็นชนิดแถวลำดับ มีสมาชิก 10
Data : Real;	ตัว ของชนิด AElement (เป็นระเบียน) สมาชิก
Key : Integer	แต่ละตัวมีเขตข้อมูลชื่อ Data และ Key
end; {AElement}	
DataArray = array [1..10] of AElement;	
var	
MyData : DataArray;	MyData เป็นตัวแปรชนิด DataArray
การอ้างถึงแถวลำดับของระเบียน	
MyData[1]. Data := 3.14159;	ค่าจริง 3.14159 เก็บในเขตข้อมูล Data ของ
MyData[10]. Key := 9999;	แถวลำดับ MyData ตัวแรก ค่า 9999 เก็บใน
	เขตข้อมูล Key ตัวสุดท้าย

แบบฝึกหัด Quick-Check

1. ข้อแตกต่างหลักระหว่างระเบียนและแถวลำดับคืออะไร ชุดใดควรใช้เก็บคำอธิบายรายกระบวนวิชาของวิชา และชุดใดควรใช้เก็บชื่อของนักศึกษาในวิชา

2. จงเขียนการประกาศชนิด อธิบายการแสดงผลของคอมพิวเตอร์ ซึ่งมี 640 สดมภ์ และ 480 แถวของจุดภาพ (pixels) เมื่อแต่ละจุดภาพมีค่าความเข้มสีเป็นแดง, เขียว และน้ำเงิน (ชนิด integer)

3. เมื่อใดเราจึงจะสามารถใช้ตัวกำหนดค่า (assignment operator) กับตัวถูกดำเนินการชนิดระเบียนได้ และเมื่อใดเราจึงจะใช้ตัวดำเนินการเท่ากัน (equality operate) ได้

4. สำหรับตัวแปร AStudent ซึ่งประกาศข้างล่างนี้ จงจัดหาข้อความสั่ง ซึ่งแสดงผลค่าเริ่มต้นของ Astudent

```
type
   StringType =  string [StringLength];
   Student    =  record
                    First, Last : StringType;
                    Age, Score : Integer;
                    Grade : Char
                 end; {Student}
var
   Astudent :    Student;
```

5. ถ้า integer ใช้เนื้อที่หน่วยเก็บสองไบต์ ตัวอักขระใช้เนื้อที่หนึ่งไบต์ และ String Type ใช้ 20 ไบต์ จงคำนวณว่า ตัวแปร AStudent จะใช้เนื้อที่หน่วยเก็บทั้งหมดกี่ไบต์

6. จงเขียนกระบวนงานแสดงผลตัวแปรชนิด Student

7. ทำไมเราใช้ข้อความสั่ง with จงบอกข้อดี และข้อไม่ดีของการใช้ข้อความสั่ง with

คำถามทบทวน (Review Questions)

1. จงเขียนการประกาศระเบียนชื่อ Subscriber ซึ่งประกอบด้วยเขตข้อมูลชื่อ Name, StreetAddress, MonthlyBill (จำนวนเงินที่สมาชิกเป็นหนี้) และชนิดของเอกสารซึ่งสมาชิกจะได้รับ (Morning, Evening หรือ Both)

2. จงเขียนโปรแกรม Pascal เพื่อใส่ข้อมูล และหลังจากนั้นให้พิมพ์ข้อมูลในระเบียน competition ซึ่งประกาศดังนี้

```
const
   StringLength  =  20;
type
   StringType    =  string [StringLength];
```

```
    OlympicEvent = record
                       Event,
                       Entrant,
                       Country : StringType;
                       Place : Integer
                   end; {OlympicEvent}
var
    Competition  :  OlympicEvent;
```

3. จงบอกที่ผิดและแก้ไขให้ถูกต้องในโปรแกรมข้างล่างนี้

```
program Report;
  type
    String15 = string[15];
    SummerHelp = record
                     Name : String15;
                     StartDate : String15;
                     HoursWorked : Real
                 end; {SurnmerHelp}
  var
    Operator : SummerHelp;
begin {Report}
  with SummerHelp do
    begin
      Name := 'Stoney Viceroy';
      StartDate := 'June 1, 2008';
      HoursWorked := 29.3
    end; {with}
  WriteLn (Operator)
end. {Report}
```

จงใช้การประกาศข้างล่างนี้สำหรับคำถามทบทวนข้อ 4 ถึงข้อ 9

```
const
   TotalEmployees  =  20;
type
   EmpRange =  1..TotalEmployees;
   Employee  =  record
                   ID : Integer;
                   Rate, Hours : Real
                end; {Employee}
   EmpArray  =  array [EmpRange] of Employee;
var
   Employees =  EmpArray;
```

4. จงเขียนฟังก์ชัน TotalCross ซึ่งจะกลับคืน ค่าจ้างสุทธิทั้งหมด (total gross pay) ซึ่งกำหนดให้เป็นข้อมูลอยู่ในแถวลำดับ Employees

5. จงอธิบายว่ามีอะไรผิด (wrong) ในส่วนของโปรแกรมข้างล่างนี้ และแก้ไขให้ถูกต้อง

```
I  := 1;
with Employee [I] do
   while I <= TotalEmployees do
      begin
         WriteLn (Hours : 12 : 2);
         I := I + 1
      end  {while}
```

6. จงอธิบายว่า มีอะไรผิดในส่วนของโปรแกรมข้างล่างนี้และแก้ไขให้ถูกต้อง

```
I := 1;
while (Employee[I].ID <> 999) and (I <= TotalEmployees) do
      I := I + 1;
```

7. จงเขียนส่วนของโปรแกรม (a frangment) ซึ่งแสดงผล ID number ของพนักงานแต่ละคนซึ่งทำงาน ระหว่าง 10 ถึง 20 ชั่วโมงต่อสัปดาห์

8. จงเขียนส่วนของโปรแกรมซึ่งแสดงผล ID number ของพนักงานคนที่มีจำนวนชั่วโมงทำงานมากที่สุด

9. จงเขียนกระบวนงานซึ่งกำหนด employee ID ให้ แล้วแสดงผลค่าจ้างของพนักงานคนนั้น