Chapter 9 Structures

- Array ในบทที่ 7 เป็นตัวแปรชนิดหนึ่งที่อนุญาตให้ใส่ข้อมูลชนิดเดียวกันหลายๆ ค่าลงในตัวแปรตัวเดียว การเข้าถึงสมาชิกแต่ละตัวของตัวแปร array ให้ตามหลังชื่อของ array ด้วย index
- ภาษาซีมีเครื่องมืออีกรูปแบบหนึ่งที่ให้มีการรวมข้อมูลหลายๆชนิดลงในตัวแปรตัวเดียวกัน โดยมีชื่อเรียกเฉพาะว่า structure
- ยกตัวอย่างเช่นเราต้องการเก็บค่าวันที่ 20/09/2009 ลงในโปรแกรม โดยปกติเราจะต้องแยกเก็บค่าวันเดือนปีในตัวแปรสามตัว เช่น int day = 20, month = 9, year = 2009 หากในโปรแกรมเดียวกันเราต้องการเก็บวันที่อีกวันหนึ่ง ก็จะต้องมีตัวแปรทั้งสามตัวอีกชุดหนึ่ง เช่น day_a, month_a, year_a
- การทำเช่นนี้ทำให้เราต้องจำกลุ่มของตัวแปรแต่ละชุดตลอดการเขียนหรือแก้ไขโปรแกรม หากเราสามารถรวมกลุ่มตัวแปร day, month, year ของวันที่เดียวกันได้จะสะดวกขึ้นมาก ซึ่งนี่เป็นเรื่องของ structure นั่นเอง

9.1 Structure เพื่อเก็บวันที่

- เราสามารถประกาศ structure ชื่อ date ที่ประกอบด้วยสามตัวแปรย่อยสำหรับเก็บวันเดือนปี โดยการประกาศ structure ดังกล่าวทำได้ดังนี้

```
struct date
{
    int month;
    int day;
    int year;
};
```

- ในการนี้ date ประกอบด้วยสมาชิกซึ่งเป็นตัวแปรชนิด integer คือ month, day, year ซึ่งจริงๆแล้วการประกาศ structure เป็นการประกาศชนิดของข้อมูลชนิดใหม่ ในที่นี้คือข้อมูลชนิด struct date และเราสามารถประกาศตัวแปรตามชนิดของข้อมูลได้ดังนี้

  struct date today;
  struct date tomorrow;

- การอ้างอิงสมาชิกแต่ละตัวในตัวแปรชนิด struct ทำได้โดยอ้างถึงชื่อของตัวแปรตามด้วยชื่อของสมาชิกในตัวแปร ยกตัวอย่างเช่น

  today.day = 25;
  today.year = 2004;
  if(today.month ==1 && today.day ==1){…}

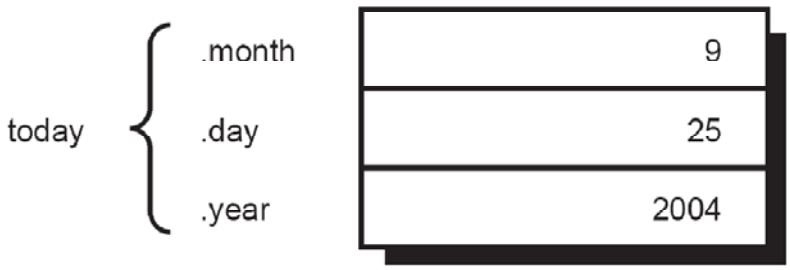
Program 9.1 Illustrating a Structure

```
// Program to illustrate a structure
#include <stdio.h>
int main(void)
{
    struct date
    {
        int month;
        int day;
        int year;
    };
    struct date today;
    today.month = 9;
    today.day = 25;
    today.year = 2004;
    printf ("Today's date is %i/%i/%.2i.\n", today.month,
today.day, today.year % 100);
    return 0;
}
```

```
Today's date is 9/25/04.
```

- คำสั่งแรกในฟังก์ชัน main ประกาศ structure ชื่อ date ประกอบด้วยสมาชิกสามตัวคือ month, day, year ในคำสั่งมา ตัวแปรชื่อ today ถูกประกาศให้เป็นชนิด struct date
- การประกาศ structure จะยังไม่มีการจองหน่วยความจำสำหรับเก็บข้อมูล แตกต่างจากการประกาศตัวแปรชนิด structure ซึ่งจะมีการจองหน่วยความจำไว้สำหรับเก็บข้อมูลที่เป็นของสมาชิกของ structure
- หลังจากตัวแปร today ถูกประกาศ โปรแกรมดำเนินการต่อโดยใส่ข้อมูลลงในสมาชิกแต่ละตัวของตัวแปร ดังรูป

```
today.month = 9;
today.day = 25;
today.year = 2004;
```



<u>ตัวอย่างการใช้งาน structure เพิ่มเติม</u>

- การใช้งานตัวแปร structure เหมือนกับการใช้งานตัวแปรปกติทุกประการ เช่น การหารสมาชิกชนิด integer ด้วย integer จะเป็นการหารแบบ integer

  century = today.year / 100 + 1;
- หรือหากต้องการเขียนโปรแกรมที่รับวันที่ปัจจุบันมาแล้วแสดงผลเป็นวันที่ของพรุ่งนี้ เราสามารถใช้ชุดคำสั่งเช่น

  tomorrow.month = today.month;
  tomorrow.day = today.day + 1;
  tomorrow.year = today.year;
- อย่างไรก็ตาม คำสั่งด้านบนจะไม่ถูกต้องในกรณีที่ วันที่ปัจจุบันเป็นวันสุดท้ายของเดือน หรือ วันที่ปัจจุบันเป็นวันสุดท้ายของปี
- เพื่อที่จะตรวจสอบว่าวันที่ปัจจุบันเป็นวันสุดท้ายของเดือนหรือไม่เราสามารถตรวจสอบได้จากอาร์เรย์ที่เก็บจำนวนวันที่ในแต่ละเดือน

  int daysPerMonth[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
  จากอาร์เรย์นี้ daysPerMonth[0] คือจำนวนวันในเดือนมกราคม

  daysPerMonth[1] คือจำนวนวันในเดือนกุมภาพันธ์ ...
- โปรแกรมจะต้องตรวจสอบด้วยว่าเป็นวันสุดท้ายของปีหรือไม่ เพื่อที่จะเพิ่มปีขึ้นหนึ่งปี

Program 9.2 Determining Tomorrow's Date

```
// Program to determine tomorrow's date
#include <stdio.h>

int main (void)
{
    struct date
    {
        int month;
        int day;
        int year;
    };
    struct date today, tomorrow;
    const int daysPerMonth[12] = { 31, 28, 31, 30, 31, 30,
31, 31, 30, 31, 30, 31 };

    printf("Enter today's date (mm dd yyyy): ");
    scanf("%i%i%i", &today.month, &today.day, &today.year);

    if( today.day != daysPerMonth[today.month - 1] )
    {
        tomorrow.day = today.day + 1;
        tomorrow.month = today.month;
        tomorrow.year = today.year;
    }
    else if( today.month == 12 )
    { // end of year
        tomorrow.day = 1;
        tomorrow.month = 1;
        tomorrow.year = today.year + 1;
    }
    else { // end of month
        tomorrow.day = 1;
        tomorrow.month = today.month + 1;
        tomorrow.year = today.year;
    }
    printf ("Tomorrow's date is %i/%i/%.2i.\n", tomorrow.month,
        tomorrow.day, tomorrow.year % 100);
    return 0;
}
```

```
Enter today's date (mm dd yyyy): 12 17 2004
Tomorrow's date is 12/18/04.
```

- โปรแกรมประกาศ struct date และประกาศตัวแปรชื่อ today และ tomorrow ให้เป็นชนิด struct date จากนั้นโปรแกรมรับเอาวันที่ปัจจุบันจากผู้ใช้มาเก็บไว้ใน today.month, today.day, และ today.year
- จากนั้นในเงื่อนไขแรกโปรแกรมตรวจสอบว่าเป็นวันสุดท้ายของเดือนหรือไม่ หากไม่ใช่ ให้บวกหนึ่งเข้ากับวันปัจจุบันและให้เดือนและปีไม่เปลี่ยนแปลง
- หากเป็นวันสุดท้ายของเดือน เงื่อนไขถัดมาจะตรวจสอบว่าเป็นวันสุดท้ายของปีด้วยหรือไม่ หากใช่ ก็จะให้ผลลัพธ์เป็นวันที่ 1 เดือน 1 ของปีถัดไป หากไม่ใช่จะให้ผลลัพธ์เป็นวันที่ 1 ของเดือนใหม่
- สังเกตว่าโปรแกรมนี้ไม่มีการตรวจสอบ leap year หรือปีที่เดือนกุมภาพันธ์มี 29 วัน ซึ่งจะได้แก้ไขในโปรแกรมถัดไป

<u>9.2 การส่งข้อมูลชนิด structure เข้าไปในฟังก์ชัน</u>

- ในตัวอย่างถัดไปจะแก้ไขข้อผิดพลาดในการตรวจสอบ leap year จากโปรแกรมที่ผ่านมา
- โดยจะเพิ่มฟังก์ชันชื่อ numberOfDays เพื่อหาจำนวนของวันในเดือนที่กำหนด และจะมีการตรวจสอบ leap year ด้วย ส่วนในฟังก์ชัน main จะเปลี่ยนเฉพาะส่วนของเงื่อนไขแรกที่เทียบวันที่กับจำนวนวันที่จากตาราง เป็นการเทียบวันที่กับจำนวนวันที่คืนมาโดยฟังก์ชัน numberOfdays
- ให้สังเกตการส่งค่าเข้าไปในฟังก์ชัน numberOfDays

Program 9.3 Revising the Program to Determine Tomorrow's Date

```
// Program to determine tomorrow's date
#include <stdio.h>
#include <stdbool.h>

struct date
{
    int month;
    int day;
    int year;
};

int main (void)
{
    struct date today, tomorrow;
    int numberOfDays(struct date d);

    printf("Enter today's date (mm dd yyyy): ");
    scanf("%i%i%i", &today.month, &today.day, &today.year);

    if( today.day != numberOfDays(today) )
    {
        tomorrow.day = today.day + 1;
        tomorrow.month = today.month;
        tomorrow.year = today.year;
    }
    else if( today.month == 12 )
    { // end of year
        tomorrow.day = 1;
        tomorrow.month = 1;
        tomorrow.year = today.year + 1;
    }
    else
    { // end of month
        tomorrow.day = 1;
        tomorrow.month = today.month + 1;
        tomorrow.year = today.year;
    }
    printf ("Tomorrow's date is %i/%i/%.2i.\n",tomorrow.month,
        tomorrow.day, tomorrow.year % 100);
    return 0;
}
```

```
// Function to find the number of days in a month
int numberOfDays(struct date d)
{
    int days;
    bool isLeapYear(struct date d);
    const int daysPerMonth[12] =
        { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
    if( isLeapYear (d) == true && d.month == 2 )
        days = 29;
    else
        days = daysPerMonth[d.month - 1];
    return days;
}

// Function to determine if it's a leap year
bool isLeapYear(struct date d)
{
    bool leapYearFlag;
    if((d.year%4 == 0 && d.year%100 != 0)|| d.year%400 == 0 )
        leapYearFlag = true; // It's a leap year
    else
        leapYearFlag = false; // Not a leap year
    return leapYearFlag;
}
```

```
Enter today's date (mm dd yyyy): 2 28 2004
Tomorrow's date is 2/29/04.
```

```
Enter today's date (mm dd yyyy): 2 28 2005
Tomorrow's date is 3/1/05.
```

- สังเกตว่า struct date ถูกประกาศไว้นอกฟังก์ชันทั้งหมด ซึ่งทำให้ชนิดข้อมูลนี้เป็นที่รู้จักและใช้ได้ทั้งโปรแกรม หรือเป็น global structure ซึ่งคล้ายกับการประกาศตัวแปรแบบ global และ local นั่นเอง
- ในฟังก์ชัน main มีการประกาศฟังก์ชัน int numberOfDays(struct date d); เพื่อบ่งบอกว่าฟังก์ชันนี้ส่งเลขจำนวนเต็ม (จำนวนวัน) คืนมาและรับ argument หนึ่งตัวเป็นชนิด struct date (วันที่ปัจจุบัน)

- ในเงื่อนไขแรก today.day != numberOfDays(today) เป็นการเปรียบเทียบวันที่ปัจจุบันกับจำนวนวันในเดือนนั้น โดยจำนวนวันเป็นค่าที่คืนกลับมาจากฟังก์ชั่น numberOfDays
- ฟังก์ชัน numberOfDays รับ argument เป็นข้อมูลชนิด struct date และการเปลี่ยนแปลงข้อมูลภายในฟังก์ชันไม่มีผลกับตัวแปร struct date ที่ส่งค่าให้
- ฟังก์ชัน numberOfDays ตรวจสอบว่าเป็นปี leap year และเป็นเดือนกุมภาพันธ์หรือไม่ หากใช่ จะส่งจำนวนวันคืนมาเป็น 29 วัน กรณีอื่นนอกจากนั้นฟังก์ชันจะดูจำนวนวันจากตารางที่กำหนดไว้ในตัวแปร daysPerMonth
- ฟังก์ชัน isLeapYear ใช้ตรวจสอบ leap year คือปี คศ ที่หารด้วยสี่ลงตัวแต่หารด้วย 100 ไม่ลงตัว หรือ ปีที่หารด้วย 400 ลงตัว หากเป็น leap year ฟังก์ชันจะคืนค่าเป็น true และถ้าไม่ใช่ ฟังก์ชันจะคืนค่าเป็น false
- โปรแกรมถัดไปจะแยกการคำนวณวันรุ่งขึ้น ออกจาก main มาเป็นฟังก์ชันชื่อ dateUpdate โดยฟังก์ชันนี้จะรับวันที่ปัจจุบันเข้าไป เป็นชนิด struct date และคืนค่าวันรุ่งขึ้นออกมาเป็นชนิด struct date เช่นกัน

Program 9.4 Revising the Program to Determine Tomorrow's Date

```
// Program to determine tomorrow's date
#include <stdio.h>
#include <stdbool.h>

struct date
{
    int month;
    int day;
    int year;
};

// Function to determine if it's a leap year
bool isLeapYear (struct date d)
{
    bool leapYearFlag;
    if ((d.year%4 == 0 && d.year%100 != 0) || d.year%400 == 0 )
        leapYearFlag = true; // It's a leap year
    else
        leapYearFlag = false; // Not a leap year
    return leapYearFlag;
}

// Function to find the number of days in a month
int numberOfDays (struct date d)
{
    int days;
    const int daysPerMonth[12]
        ={31,28,31,30,31,30,31,31,30,31,30,31};
    if( isLeapYear(d) == true && d.month == 2 )
        days = 29;
    else
        days = daysPerMonth[d.month - 1];
    return days;
}

// Function to calculate tomorrow's date
struct date dateUpdate(struct date today)
{
    struct date tomorrow;
    if( today.day != numberOfDays (today) )
    {
        tomorrow.day = today.day + 1;
        tomorrow.month = today.month;
        tomorrow.year = today.year;
```

```
    }
    else if ( today.month == 12 )
    { // end of year
        tomorrow.day = 1;
        tomorrow.month = 1;
        tomorrow.year = today.year + 1;
    }
    else
    { // end of month
        tomorrow.day = 1;
        tomorrow.month = today.month + 1;
        tomorrow.year = today.year;
    }
    return tomorrow;
}

int main (void)
{
    struct date thisDay, nextDay;
    printf("Enter today's date (mm dd yyyy): ");
    scanf("%i%i%i",&thisDay.month, &thisDay.day, &thisDay.year);
    nextDay = dateUpdate(thisDay);
    printf("Tomorrow's date is %i/%i/%.2i.\n",nextDay.month,
        nextDay.day, nextDay.year % 100);
    return 0;
}
```

```
Enter today's date (mm dd yyyy): 2 28 2008
Tomorrow's date is 2/29/08.
```

```
Enter today's date (mm dd yyyy): 2 22 2005
Tomorrow's date is 2/23/05.
```

- จากโปรแกม ฟังก์ชัน main เรียกใช้ฟังก์ชัน dateUpdate ซึ่งเรียกใช้ฟังก์ชัน numberOfDays ที่เรียกฟังก์ชัน isLeapYear

การใช้ structure เพื่อเก็บเวลา

- เช่นเดียวกับวันที่ เราสามารถสร้าง structure เพื่อเก็บค่าเวลาซึ่งประกอบไปด้วย ชั่วโมง นาที และวินาทีได้ ดังตัวอย่าง

```
struct time
{
     int hour;
     int minutes;
     int seconds;
}
```

- การเขียนโปรแกรมเพื่อเพิ่มค่าเวลาครั้งละหนึ่งวินาที
    - ถ้าวินาทีครบ 60 ปรับให้เป็น 0 และเพิ่มนาทีขึ้นหนึ่ง
    - ถ้านาทีครบ 60 ปรับให้เป็น 0 และเพิ่มชั่วโมงขึ้นหนึ่ง
    - ถ้าชั่วโมงครบ 24 ปรับให้เป็น 0
- เขียนเป็นโปรแกรมได้ดังนี้

Program 9.5 Updating the Time by One Second

```
// Program to update the time by one second
#include <stdio.h>
struct time
{
     int hour;
     int minutes;
     int seconds;
};
```

```
int main (void)
{
     struct time timeUpdate(struct time now);
     struct time currentTime, nextTime;

     printf("Enter the time (hh:mm:ss): ");
     scanf("%i:%i:%i", &currentTime.hour, &currentTime.minutes,
&currentTime.seconds);

     nextTime = timeUpdate(currentTime);

     printf("Updated time is %.2i:%.2i:%.2i\n", nextTime.hour,
nextTime.minutes, nextTime.seconds );
     return 0;
}

// Function to update the time by one second
struct time timeUpdate(struct time now)
{
     ++now.seconds;
     if( now.seconds == 60 )
     {
          now.seconds = 0;
          ++now.minutes;
          if( now.minutes == 60 )
          {
               now.minutes = 0;
               ++now.hour;
               if( now.hour == 24 )
                    now.hour = 0;
          }
     }
     return now;
}
```

```
Enter the time (hh:mm:ss): 12:23:55
Updated time is 12:23:56
```

```
Enter the time (hh:mm:ss): 16:12:59
Updated time is 16:13:00
```

```
Enter the time (hh:mm:ss): 23:59:59
Updated time is 00:00:00
```

- โปรแกรมรับค่าเวลาจากผู้ใช้ด้วยคำสั่ง scanf โดยใช้รูปแบบ “%i:%i:%i” สังเกตว่ามีเครื่องหมาย “:” คั่น หมายความว่าผู้ใช้จะต้องใส่ค่าเวลาในรูปแบบเดียวกัน เช่น 16:12:59 และไม่ใช่ 16 12 59 หรือ 16-12-59
- จากนั้นโปรแกรมเรียกฟังก์ชัน timeUpdate โดยส่ง currentTime เป็น argument ให้ และค่าที่คืนจากฟังก์ชันนี้ถูกใส่ให้กับ nextTime
- ในฟังก์ชัน timeUpdate เริ่มทำงานด้วยการเพิ่มค่าเวลาในวินาทีขึ้นหนึ่งวินาที จากนั้นตรวจสอบว่าถึง 60 วินาทีหรือยัง ถ้าหากถึงแล้วให้ปรับวินาทีกลับเป็น 0 แล้วเพิ่มขึ้นหนึ่งนาที จากนั้นตรวจสอบว่าถึง 60 นาทีหรือยัง ถ้าถึงแล้วให้ปรับนาทีเป็น 0 แล้วเพิ่มชั่วโมงขึ้นหนึ่ง จากนั้นตรวจสอบว่าถึง 24 ชั่วโมงหรือยัง ถ้าถึงแล้วให้ปรับชั่วโมงเป็น 0

9.3 การใส่ค่าเริ่มต้นให้ structure

- การใส่ค่าเริ่มต้นให้ structure ตอนประกาศตัวแปร คล้ายกับการใส่ค่าเริ่มต้นให้ array กล่าวคือค่าเริ่มต้นจะถูกใส่รวมอยู่ในเครื่องหมายวงเล็บปีกกา และสมาชิกแต่ละตัวแยกด้วยลูกน้ำ
- การใส่ค่าเริ่มต้นให้ตัวแปร today เป็น August 20, 2009 ทำได้ดังตัวอย่าง

  struct date today = {8, 20, 2009};
- หรือการใส่ค่าเวลา 12:29:55

  struct time now = {12, 29, 55};

- เราสามารถใส่ค่าเริ่มต้นเฉพาะสมาชิกบางตัวเช่น

  struct time time1 = {12, 29};

  หรือ

  struct time time1 = {.hour = 12, .minutes = 29};

  จะหมายถึงเวลา 12 นาฬิกา 29 นาที และวินาทีเป็นข้อมูลขยะในหน่วยความจำ

9.4 Array ของ structure

- เราได้เห็นประโยชน์ของ structure ในการจับกลุ่มตัวแปรหลายตัวที่เกี่ยวข้องกันเอาไว้อยู่ตัวแปรตัวเดียวกันแล้ว เช่น struct time ที่เก็บเอาตัวแปรสามตัวเข้าไว้ด้วยกัน
- หากมีเวลาสิบค่าที่แตกต่างกันและต้องการเก็บในตัวแปรชนิด struct time เราสามารสร้างตัวแปรนั้นขึ้นเป็น array ได้ เช่น struct time experiments[10]; เป็นการสร้าง array ชื่อ experiments ที่ประกอบด้วยสมาชิกสิบตัว โดยสมาชิกแต่ละตัวสามารถเก็บชั่วโมง นาที วินาที ไว้ในรูปแบบของ structure
- หรือ struct date birthdays[15]; สร้าง array ชนิด struct date ที่ประกอบด้วยสมาชิก 15 ตัว การเข้าถึงสมาชิกแต่ละตัวทำได้ดังตัวอย่าง

  birthdays[0].month = 8;
  birthdays[0].day = 8;
  birthdays[0].year = 1991;
- การส่งสมาชิกแต่ละตัวเข้าไปในฟังก์ชันซึ่งรับ argument ในรูปแบบ structure ก็ทำได้เช่นเดียวกันเช่น

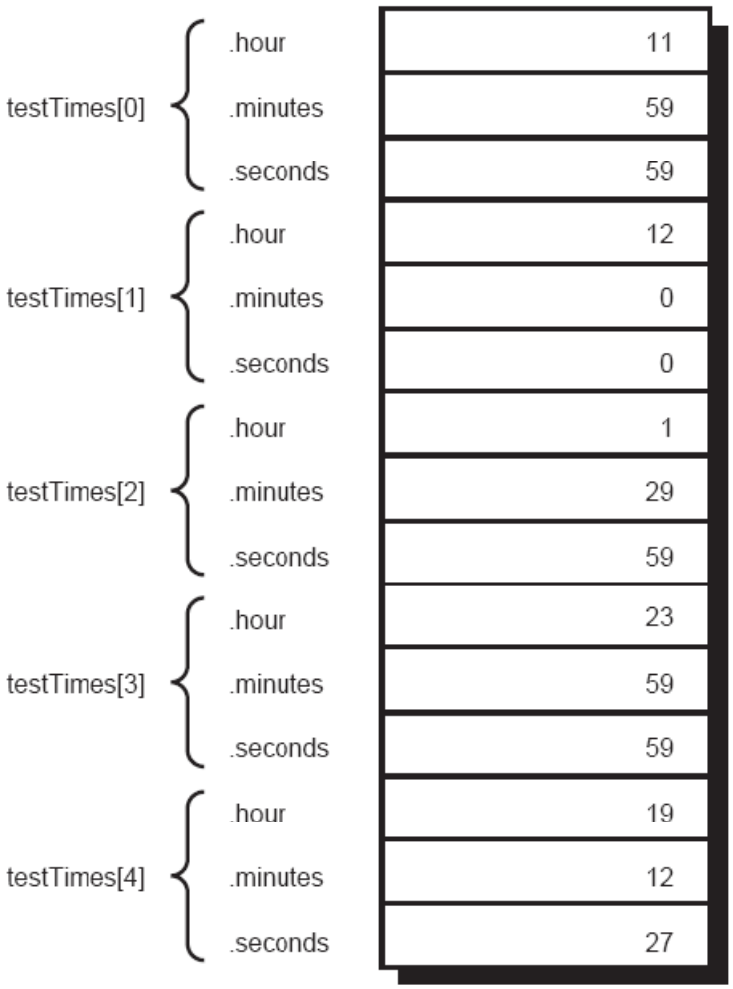
  checkTime(experiments[4]);

- การใส่ค่าเริ่มต้นให้กับ array ที่มีสมาชิกเป็น structure ทำได้ดังตัวอย่าง

  struct time runtime[5] = {{12, 0, 0}, {12, 30, 0}, {13, 15, 0}};
  เป็นการตั้งเวลาเริ่มต้นให้กับตัวแปร runtime[0] เป็น 12:00:00, runtime[1] เป็น 12:30:00 และ runtime[2] เป็น 13:15:00

- หรือ struct time testTimes[5] =

  {{11, 59, 59}, {12, 0, 0}, {1, 29, 59}, {23, 59, 59}, {19, 12, 27}};
  จะถูกเก็บในหน่วยความจำดังแสดงในรูป



- โปรแกรมต่อไปนี้จะสร้าง array ชื่อ testTimes และส่งเป็น argument ของฟังก์ชัน timeUpdate (ไม่ได้เขียนรวมไว้ในโปรแกรมนี้)

Program 9.6 Illustrating Arrays of Structures

```
// Program to illustrate arrays of structures
#include <stdio.h>

struct time
{
    int hour;
    int minutes;
    int seconds;
};

int main (void)
{
    struct time timeUpdate(struct time now);
    struct time testTimes[5] =
    {{11,59,59}, {12,0,0}, {1,29,59}, {23,59,59 }, {19,12,27}};
    int i;

    for( i = 0; i < 5; ++i )
    {
        printf("Time is %.2i:%.2i:%.2i", testTimes[i].hour,
            testTimes[i].minutes, testTimes[i].seconds);

        testTimes[i] = timeUpdate(testTimes[i]);

        printf(" ...one second later it's %.2i:%.2i:%.2i\n",
            testTimes[i].hour, testTimes[i].minutes,
            testTimes[i].seconds);
    }
    return 0;
}
// Include the timeUpdate function from Program 9.5 here *****
```

```
Time is 11:59:59 ...one second later it's 12:00:00
Time is 12:00:00 ...one second later it's 12:00:01
Time is 01:29:59 ...one second later it's 01:30:00
Time is 23:59:59 ...one second later it's 00:00:00
Time is 19:12:27 ...one second later it's 19:12:28
```

9.5 Structure ที่มีสมาชิกเป็น Structure

- เราสามารถสร้างตัวแปรชนิด structure ที่ประกอบด้วยสมาชิกที่เป็น structure
- อย่างเช่นเราได้รวม hour, minutes, และ seconds ไว้ใน struct time และรวม month, day, year ไว้ใน struct date ในการทำงานบางอย่างเช่นการบันทึกเหตุการณ์ต่างๆ เราอาจต้องการรวมกลุ่ม date และ time เอาไว้ด้วยกันเพื่อความสะดวกในการอ้างอิง
- ดังนั้นเราจะสร้าง structure ชนิดใหม่ ชื่อ dateAndTime ซึ่งประกอบด้วยสมาชิกคือ date และ time

```
struct dateAndTime
{
        struct date sdate;
        struct time stime;
};
```

ในที่นี้สมาชิกตัวแรกเป็นชนิด struct date ชื่อ sdate และ สมาชิกตัวที่สองเป็นชนิด struct time ชื่อ stime ซึ่งทั้ง struct date และ struct time จะต้องถูกประกาศไว้ก่อนหน้าแล้ว

- หลังจากการประกาศชนิดตัวแปร struct dateAndTime เราสามารถสร้างตัวแปรเป็นชนิดนั้นได้ ดังเช่น struct dateAndTime event;
- การเข้าถึงข้อมูลวันที่ ทำได้โดยใช้ event.sdate และสามารถส่งผ่านเข้าไปยังฟังก์ชันได้ตามปกติ เช่น event.sdate = dateUpdate(event.sdate);

- เช่นเดียวกัน การเข้าถึงข้อมูลเวลาใช้ event.stime และการส่งข้อมูลไปยังฟังก์ชันเป็น event.stime = timeUpdate(event.stime);
- การอ้างถึงสมาชิกเจาะจงตัวเช่น event.sdate.month = 10; หรือ ++event.stime.seconds;
- การใส่ค่าเริ่มต้น เช่น

  struct dateAndTime event = {{2, 1, 2004}, {3, 30, 0}};
  เป็นการตั้งวันที่เป็น February 1, 2004 เวลา 3:30:00 นาฬิกา

9.6 Structure ที่มีสมาชิกเป็น Array

- Structure มีสมาชิกเป็น array ได้ เช่นเราต้องการประกาศ structure ชนิดใหม่ชื่อ month ที่มีสมาชิกคือจำนวนวันในเดือนนั้น และตัวอักษรสามตัวย่อชื่อเดือนนั้น สามารถประกาศได้ดังนี้

```
struct month
{
        int numberOfDays;
        char name[3];
};
```

  จากนั้นประกาศตัวแปรชนิด struct month ดังนี้

  struct month aMonth;
- การเข้าถึงสมาชิกแต่ละตัวเป็นดังนี้

```
aMonth.numberOfDays = 31;
aMonth.name[0] = 'J';
aMonth.name[1] = 'a';
```

aMonth.name[2] = ‘n’;

- หรือจะใส่ค่าเริ่มต้นให้ในตอนประกาศตัวแปร ดังนี้

  struct month aMonth = {31, {‘J’, ‘a’, ‘n’}};

- หากมี 12 เดือนก็สามารถประกาศตัวแปรชนิด struct month ให้เป็น array ได้ เช่น struct month months[12];

- โปรแกรมถัดไปเป็นการใช้งานตัวแปรชนิด struct month โดยใส่ค่าเริ่มต้นให้กับตัวแปร และแสดงค่าเหล่านั้นออกทางหน้าจอ

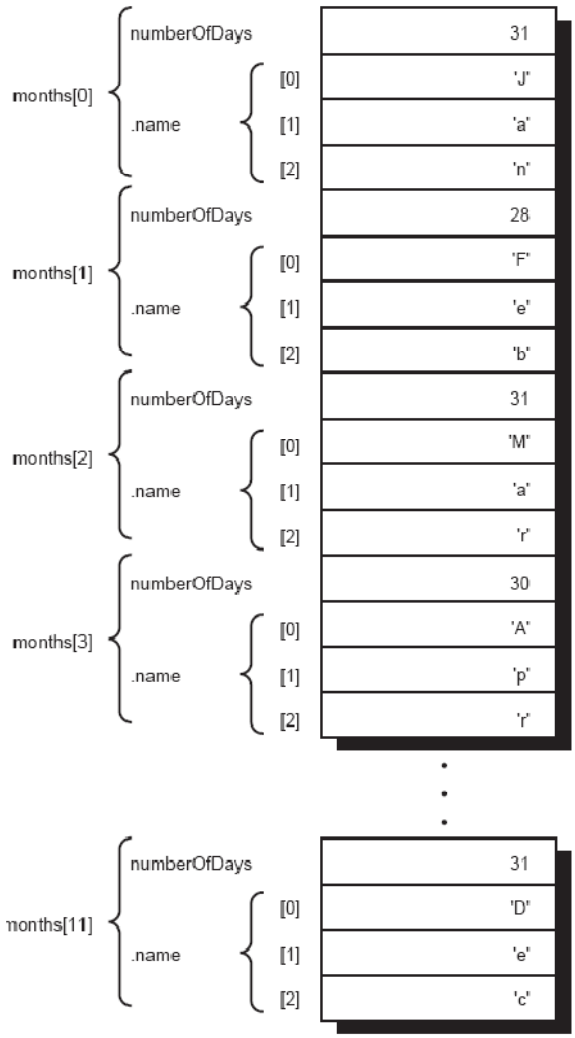
Program 9.7 Illustrating Structures and Arrays

```
// Program to illustrate structures and arrays
#include <stdio.h>
int main (void)
{
    int i;
    struct month
    {
        int numberOfDays;
        char name[3];
    };
    const struct month months[12] =
      { { 31, {'J', 'a', 'n'} }, { 28, {'F', 'e', 'b'} },
        { 31, {'M', 'a', 'r'} }, { 30, {'A', 'p', 'r'} },
        { 31, {'M', 'a', 'y'} }, { 30, {'J', 'u', 'n'} },
        { 31, {'J', 'u', 'l'} }, { 31, {'A', 'u', 'g'} },
        { 30, {'S', 'e', 'p'} }, { 31, {'O', 'c', 't'} },
        { 30, {'N', 'o', 'v'} }, { 31, {'D', 'e', 'c'} } };

    printf ("Month Number of Days\n");
    printf ("----- --------------\n");

    for ( i = 0; i < 12; ++i )
        printf (" %c%c%c %i\n",
            months[i].name[0], months[i].name[1],
            months[i].name[2], months[i].numberOfDays);
    return 0;
}
```

```
Month      Number of Days
-----      ---------------
Jan             31
Feb             28
Mar             31
Apr             30
May             31
Jun             30
Jul             31
Aug             31
Sep             30
Oct             31
Nov             30
Dec             31
```



9.7 การประกาศตัวแปร structure ในแบบอื่น

- ตัวแปร structure สามารถประกาศไว้พร้อมกับการประกาศ structure ได้เลย

```
struct date
{
        int month;
        int day;
        int year;
} todaysDate, purchaseDate;
```

เป็นการประกาศ structure ชนิดใหม่ และประกาศตัวแปรสองตัวไว้ด้วย

- ในการนี้เราสามารถใส่ค่าเริ่มต้นให้ได้เช่นกัน

```
struct date
{
        int month;
        int day;
        int year;
} todaysDate = {8, 20, 2009};
```