# แผนบริหารการสอนประจำบทที่ 8

**หัวข้อเนื้อหา**

- การประกาศข้อมูล
- การอ้างใช้ข้อมูลที่ประกาศไว้
- การอ้างตำแหน่งของข้อมูล
- การประกาศข้อมูลสำหรับการใช้บริการของ DOS หมายเลข 09h และ 0Ah
- การใช้บริการของ DOS หมายเลข 0Ah : การอ่านข้อความ
- การใช้บริการของ DOS หมายเลข 09h : การอ่านพิมพ์ข้อความ

**วัตถุประสงค์เชิงพฤติกรรม**

- เข้าใจรูปแบบการประกาศข้อมูลหรือตัวแปรในโปรแกรมภาษาแอสเซมบลี
- สามารถและเข้าใจการอ้างใช้ข้อมูลหรือตัวแปรที่ประกาศไว้
- สามารถและเข้าใจฟังก์ชั่นหมายเลข 09h และ 0Ah ของ DOS ว่าเป็นฟังก์ชั่นที่ทำงานอย่างไร

**วิธีสอนและกิจกรรมการเรียนการสอน**

- บรรยาย
- สืบเสาะหาความรู้
- ค้นคว้าเพิ่มเติม
- ตอบคำถาม

**สื่อการเรียนการสอน**

- สื่ออิเล็กทรอนิกส์
- เอกสารอ้างอิงประกอบการค้นคว้า

**การวัดผลและประเมินผล**

**ใช้วิธีการสังเกตและจดบันทึกไว้เป็นระยะ**

- สังเกตจากงานที่กำหนดให้ไปทำมาส่ง
- สังเกตจากการตอบคำถาม
- สังเกตจากการนำความรู้ไปใช้

**การประเมินผล**

**วิธีตรวจผลงานต่างๆ ที่ให้ทำ**

- ตรวจผลงานภาคปฏิบัติ
- ตรวจรายงาน
- ตรวจแบบฝึกหัด

**ใช้วิธีการออกข้อสอบข้อเขียน**

# บทที่ 8 การประกาศข้อมูล (Data Declared)

ในบทที่แล้วเราได้ศึกษาเกี่ยวกับรูปแบบของการเขียนโปรแกรมภาษาแอสเซมบลีและการประกาศเซกเมนต์แล้ว ในบทนี้เราจะศึกษาเกี่ยวกับการประกาศข้อมูลภายในเซกเมนต์ข้อมูล และการใช้บริการหมายเลข 09h และ 0Ah ของระบบปฏิบัติการ DOS ในการแสดงผลข้อความและการอ่านข้อความจากผู้ใช้

**การประกาศข้อมูล**

การประกาศข้อมูลหรือตัวแปรในโปรแกรมภาษาแอสเซมบลีนั้น ทำได้โดยประกาศจองเนื้อที่ในหน่วยความจำในเซกเมนต์ข้อมูล แล้วตั้งเลเบลของข้อมูลนั้นไว้ ในการอ้างถึงข้อมูลในหน่วยความจำตำแหน่งนั้น เราสามารถอ้างโดยใช้เลเบลที่เราประกาศไว้ได้ ดังนั้นการประกาศตัวแปรหรือข้อมูลนั้นจะมีลักษณะเช่นเดียวกับการประกาศเลเบลนั่นเอง

**คำสั่งเทียมในการประกาศข้อมูล**

คำสั่งเทียมที่เราใช้ในการประกาศข้อมูลมีหลายคำสั่ง ดังตารางที่ 8.1 คำสั่งเทียมเหล่านี้ใช้ในระบุขนาดในการจองหน่วยความจำ

**ตารางที่ 8.1 คำสั่งเทียมสำหรับการระบุขนาดข้อมูลในการจองหน่วยความจำ**

| คำสั่งเทียม | ความหมาย |
|---|---|
| DB | Define Byte : ประกาศจองข้อมูลให้มีขนาดหน่วยละ 1 ไบต์ |
| DW | Define Word : ประกาศจองข้อมูลให้มีขนาดหน่วยละ 1 เวิร์ด ( 2 ไบต์) |
| DD | Define Doubleword : ประกาศจองข้อมูลให้มีขนาดหน่วยละ 2 เวิร์ด |
| DQ | Define Quadword : ประกาศจองข้อมูลให้มีขนาดหน่วยละ 4 เวิร์ด |
| DT | Define Ten Bytes : ประกาศจองข้อมูลให้มีขนาดหน่วยละ 10 ไบต์ |

ในการประกาศจองข้อมูลนี้จะทำให้ assembler กันเนื้อที่ในเซกเมนต์นั้นตามข้อมูลที่ระบุตามหลังคำสั่งเทียมเหล่านี้ โดยจะกันหน่วยความจำที่มีขนาดของแต่ละหน่วยตามที่ระบุในคำสั่ง

**รูปแบบของการประกาศข้อมูล**

ในการประกาศข้อมูล (ตัวแปร) เรามักประกาศในเซกเมนต์ข้อมูล โดยเราจะระบุชื่อของตัวแปรนั้น พร้อมทั้งคำสั่งเทียมที่ใช้ระบุขนาดของข้อมูล จากนั้นเราจะระบุข้อมูลต่าง ๆ ที่จะใช้ตำแหน่งที่จะจองนั้น รูปแบบในการระบุเป็นดังนี้

*variable_name* Dx *data*

**ส่วนของโปรแกรมที่ 8.1** การประกาศข้อมูล

**ตัวอย่างการประกาศข้อมูล**

จากการประกาศข้อมูลในส่วนของโปรแกรมที่ 8.2 จะมีการจัดสรรเนื้อที่ในหน่วยความจำดังรูปที่ 8.1 สังเกตว่าในการประกาศ data1 กับ data2 นั้นการระบุข้อมูลเหมือนกันแต่ขนาดของข้อมูลต่างกัน ทำให้การจองเนื้อที่ในหน่วยความจำแตกต่างกันด้วย

```
dseg    segment
data1   db      1,2
data2   dw      1,2
data3   db      'Hi',10,13
data4   dd      1234h
dseg    ends
```

**ส่วนของโปรแกรมที่ 8.2** ตัวอย่างการประกาศข้อมูล

| | | |
|---|---|---|
| DS:00h | 01h | data1 |
| DS:01h | 02h | |
| DS:02h | 01h | data2 |
| DS:03h | 00h | |
| DS:04h | 02h | |
| DS:05h | 00h | |
| DS:06h | 48h | data3 |
| 07h | 69h | |
| 08h | 0Ah | |
| 09h | 0Ch | |
| 0Ah | 34h | data4 |
| 0Bh | 12h | |
| 0Ch | 00h | |
| 0Eh | 00h | |

**รูปที่ 8.1** การจัดเรียงข้อมูลในหน่วยความจำจากการประกาศในส่วนของโปรแกรมที่ 8.2

**การระบุไม่ระบุค่าของข้อมูลที่จองเนื้อที่**

เราสามารถประกาศจองหน่วยความจำโดยไม่ระบุค่าเริ่มต้นได้โดยการระบุค่าเป็น '?' ดังเช่นในส่วนของโปรแกรมที่ 8.3 จะมีการจองเนื้อที่ไว้แต่ไม่มีการกำหนดค่าเริ่มต้น

```
data5    db      ?
data6    dw      ?
```

**ส่วนของโปรแกรมที่ 8.3** การใช้จองหน่วยความจำโดยไม่ระบุค่าเริ่มต้น

**การประกาศข้อมูลที่ซ้ำกัน**

เราสามารถใช้**คำสั่งเทียม dup** เพื่อบอกการซ้ำกันของข้อมูลได้. รูปแบบของคำสั่งเทียม dup มีดังนี้

*count* dup (*value*)

ตัวอย่างของการประกาศที่ใช้คำสั่งเทียม dup ดังเช่นในส่วนของโปรแกรมที่ 8.4

```
data7    db      10 dup (0)
data8    db      5 dup (4 dup (0))
data9    dw      5 dup (1, 2, 3 dup (4))
data10   db      20 dup (?)
```

**ส่วนของโปรแกรมที่ 8.4** การใช้คำสั่งเทียม dup

Assembler จะจองหน่วยความจำขนาด 10 ไบต์ ที่มีค่าเป็น 0 และจะให้เลเบล data7 ชี้ไปที่ตำแหน่งเริ่มต้นของข้อมูลนี้. ในส่วนของ data8 จะเป็นข้อมูลแบบไบต์จำนวน 4x5 ไบต์ ที่มีค่าเท่ากับ 0 เช่นเดียวกัน สังเกตว่าภายในเครื่องหมายวงเล็บของคำสั่งเทียม dup เราสามารถใส่ข้อมูลได้หลายค่า รวมทั้งกำหนดค่าแบบซ้ำกันโดยใช้คำสั่ง dup อีกได้ ดังเช่นตัวแปร data9 ในตัวแปร data10 เป็นการประกาศจองหน่วยความจำไว้โดยไม่ระบุค่าเริ่มต้น

**การอ้างใช้ข้อมูลที่ประกาศไว้**

ในการอ้างใช้ข้อมูลหรือตัวแปรที่ประกาศไว้ เราสามารถอ้างโดยใช้ชื่อของเลเบลที่ประกาศไว้ได้ Assembler จะจัดการนำตำแหน่งของข้อมูลนั้นมาแทนค่าให้โดยอัตโนมัติ เรายังสามารถอ้างค่าในหน่วยความจำโดยอ้างสัมพัทธ์กับเลเบลที่เรากำหนดขึ้นได้ ส่วนของโปรแกรมที่ 8.5 เป็นโปรแกรมที่อ้างใช้ค่าของตัวแปรที่เรากำหนดในส่วนของโปรแกรมที่ 8.2 โดยหลังจากการทำงานของโปรแกรมค่าในหน่วยความจำจะเปลี่ยนไปตามรูปที่ 8.2

| | | |
|---|---|---|
| DS:00h | 00h | data1 |
| DS:01h | 22h | |
| DS:02h | 01h | data2 |
| DS:03h | 00h | |
| DS:04h | 23h | |
| DS:05h | 11h | |
| DS:06h | 48h | data3 |
| 07h | 69h | |
| 08h | 0Ah | |
| 09h | 0Ch | |
| 0Ah | 34h | data4 |
| 0Bh | 12h | |
| 0Ch | 00h | |
| 0Eh | 00h | |

**รูปที่ 8.2** การเปลี่ยนแปลงค่าหลังการทำงานของโปรแกรมที่ 8.5

```
mov     al,data1
mov     bx,data2
mov     data1,0
mov     [data2+2],1123h
mov     data1[1],22h
mov     cl,byte ptr data4[2]
```

**ส่วนของโปรแกรมที่ 8.5** ตัวอย่างการเรียกใช้ตัวแปร

ค่าในรีจิสเตอร์ AL BX และ CL มีค่าเป็น 01h 01h และ 00h ตามลำดับ สังเกตว่าในการกำหนดค่าคงที่ให้กับตัวแปรในหน่วยความจำเรากระทำได้ทันทีโดยไม่ต้องระบุขนาด เนื่องจากในการประกาศตัวแปรเราได้ระบุกับ assembler แล้วว่าจะเป็นตัวแปรขนากเท่าใด. แต่ในกรณีที่เราต้องการจะอ้างแตกต่างจากที่เราระบุก็สามารถกระทำได้โดยต้องระบุขนาดของข้อมูลกำกับด้วย เช่นในคำสั่ง mov cl, byte ptr data4[2] เป็นการอ้างข้อมูลแบบ 8 บิต เพราะ CL เป็นรีจิสเตอร์ขนาด 8 บิต

**การอ้างตำแหน่งของข้อมูล**

เราสามารถอ้างถึงออฟเซ็ตของข้อมูลที่เราประกาศไว้ได้โดยใช้คำสั่งเทียม OFFSET ดังส่วนของโปรแกรมที่ 8.6

```
mov     bx,offset data1         ;bx = offset
mov     byte ptr [bx],10h
mov     bx,data2                ;bx = value at data2
```

**ส่วนของโปรแกรมที่ 8.6** การอ้างตำแหน่งของข้อมูล

**การอ้างตำแหน่งข้อมูลโดยคิดสัมพัทธ์กับรีจิสเตอร์ BX**

นอกจากการระบุตำแหน่งสัมพัทธ์กับเลเบลโดยใช้ค่าคงที่แล้ว เราสามารถระบุตำแหน่งของข้อมูลสัมพันธ์กับเลเบลโดยใช้ค่าจากรีจิสเตอร์ BX ได้ ตัวอย่างเช่นส่วนของโปรแกรมที่ 8.7

```
mov bx,0
mov ah,data3[bx]          ;ah=data3[0]
inc  bx
mov cl,data3[bx]          ;cl=data3[1]
mov bx,offset data3
mov dl,[bx+2]             ;dl=data3[2]
```

**ส่วนของโปรแกรมที่ 8.7** การอ้างตำแหน่งของข้อมูลสัมพัทธ์กับเลเบลโดยใช้ค่าจากรีจิสเตอร์ BX

ในคำสั่ง mov ก่อนบรรทัดที่ 5 เราอ้างหน่วยความจำโดยสัมพันธ์กับ data3 และค่าใน BX แต่ในคำสั่ง mov บรรทัดสุดท้ายของส่วนของโปรแกรมที่ 8.7 เราอ้างหน่วยความจำสัมพันธ์กับ BX ซึ่งเก็บออฟเซ็ตของ data3

**การประกาศข้อมูลสำหรับการใช้บริการของ DOS หมายเลข 09h และ 0Ah**

ฟังก์ชั่นหมายเลข 09h และ 0Ah ของ DOS เป็นฟังก์ชั่นที่ต้องมีการส่งแอดเดรสของข้อมูลในหน่วยความจำ การประกาศข้อมูลสำหรับฟังก์ชั่นหมายเลข 09h จะไม่มีความซับซ้อนมากนัก แต่สำหรับฟังก์ชั่นหมายเลข 0Ah การประกาศข้อมูลที่เหมาะสมจะทำให้เราเขียนโปรแกรมได้ง่ายมากขึ้น

**การใช้บริการของ DOS หมายเลข 09h : การพิมพ์ข้อความ**

ฟังก์ชั่นหมายเลข 09h นี้รับข้อมูลป้อนเข้าคือ :

AH = 09h

DS : DX = ตำแหน่งของหน่วยความจำของข้อมูลที่จะแสดง โดยข้อมูลนี้จะต้องจบด้วยอักขระ ‘$’

ถ้าเราต้องการพิมพ์ข้อความ “Hello world” เราสามารถประกาศข้อมูลในหน่วยความจำได้ดังนี้

```
dseg    segment
mesg    db        'Hello world',10,13,'$'
dseg    ends
```

**ส่วนของโปรแกรมที่ 8.8** ตัวอย่างการประกาศข้อมูลสำหรับการใช้บริการของ DOS หมายเลข 09h

เราสามารถสั่งแสดงข้อมูลดังกล่าวได้โดย

```
mov     ah,09h
mov     dx,offset mesg
int     21h
```

**ส่วนของโปรแกรมที่ 8.9** ตัวอย่างการการใช้บริการของ DOS หมายเลข 09h

*อักขระหมายเลข 10 (Line feed) และ 13 (Carriage Return) คือรหัสควบคุมใช้ในการสั่งให้ขึ้นบรรทัดใหม่*

**การใช้บริการของ DOS หมายเลข 0Ah : การอ่านข้อความ**

ฟังก์ชั่นนี้จะอ่านข้อความจากผู้ใช้จนกระทั่งผู้ใช้กดปุ่ม Enter โดยข้อมูลป้อนเข้าจะต้องระบุตำแหน่งของหน่วยความจำที่ใช้เก็บข้อมูล (บัฟเฟอร์) ของข้อความ ฟังก์ชั่นหมายเลข 0Ah นี้รับข้อมูลป้อนเข้าคือ

AH = 0Ah

DS : DX = ตำแหน่งของหน่วยความจำที่จะใช้เก็บข้อความ (บัฟเฟอร์)

บัฟเฟอร์จะต้องมีรูปแบบดังนี้

1. ไบต์แรกของหน่วยความจำเก็บค่าความยาวสูงสุดของข้อความที่อ่านได้ ความยาวนี้จะรวมรหัสขึ้นบรรทัดใหม่ด้วย

2. DOS จะเขียนความยาวจริงของข้อความที่อ่านเข้ามาได้ในไบต์ที่สอง

3. สำหรับไบต์ถัด ๆ ไปจะเป็นรหัสแอสกีของข้อความที่อ่านเข้ามา

การประกาศข้อมูลสำหรับการเรียกใช้ฟังก์ชั่นนี้จะสามารถประกาศได้ดังส่วนของโปรแกรมที่ 8.10

```
dseg    segment
maxlen  db      30          ;Maximum of 30 chars
msglen  db      ?           ;2nd byte contains the real length
msg     db      30 dup (?)  ;Message recieved
dseg    ends
```

**ส่วนของโปรแกรมที่ 8.10** ตัวอย่างการประกาศข้อมูลสำหรับการใช้บริการของ DOS หมายเลข 0Ah

เมื่อเราเรียกใช้บริการหมายเลข 0Ah เราจะส่งตำแหน่งของ maxlen ซึ่งเป็นตำแหน่งเริ่มต้นของบัฟเฟอร์ที่เราประกาศไปให้กับ DOS จากนั้นเราสามารถอ่านความจริงของข้อความที่อ่านมาได้ทางตัวแปร msglen ตัวอย่างโปรแกรมที่ 8.11 แสดงการใช้งานบริการหมายเลข 0Ah ในการอ่านข้อความและแสดงข้อความนั้นออกมาโดยใช้บริการหมายเลข 09h

ในการรับข้อความนั้นบริการหมายเลข 0Ah จะเก็บอักขระขึ้นบรรทัดใหม่ให้ด้วย ดังนั้นเราจะต้องเพื่อขนาดบัฟเฟอร์ที่จะให้เก็บข้อความไว้ 1 ไบต์ด้วย แต่ในการคืนค่าความยาวของข้อความมาให้ บริการหมายเลข 0Ah นี้จะใส่ความยาวที่ไม่รวมอักขระขึ้นบรรทัดใหม่นี้ เมื่อเรารับข้อความเสร็จแล้ว เคอร์เซอร์จะอยู่ที่ต้นบรรทัดที่เราป้อนข้อความนั้น ดังนั้นถ้าเราพิมพ์ข้อความเดิมซ้ำไปอีกครั้งจะทำให้ข้อความทับกันและจะไม่ทราบว่ามีการพิมพ์ข้อความออกมาอย่างถูกต้องหรือไม่ ดังนั้นเราจึงใช้ฟังก์ชั่นหมายเลข 09h สั่งพิมพ์ชุดอักขระสำหรับการขึ้นบรรทัดใหม่ก่อนที่จะสั่งพิมพ์ข้อความที่รับมา

ข้อความที่สั่งพิมพ์ด้วยบริการหมายเลข 09h จะต้องจบด้วยอักขระ ‘$’ ดังนั้นเราจึงต้องกำหนดค่าในไบต์สุดท้ายของข้อความที่รับมาด้วยอักขระ ‘$’ โปรแกรมนี้จะทำงานผิดพลาดถ้าภายในข้อความมีเครื่องหมาย ‘$’ อยู่ด้วย

```
;
; display string
;
dseg    segment
;string buffer
maxlen  db      30              ;29 chars + 1 return
msglen  db      ?
msg             db      30 dup (?)      ;29 chars + 1 return
;newline string
newline         db      10,13,'$'
dseg    ends

sseg    segment         stack
        db      100h dup (?)
sseg    ends

cseg    segment
        assume          cs:cseg,ds:dseg,ss:sseg
start:
        mov     ax,dseg                 ;set DS
        mov     ds,ax

        mov     ah,0Ah                  ;read string
        mov     dx,offset maxlen
        int     21h

        mov     ah,09h                  ;newline
        mov     dx,offset newline
        int     21h

        mov     bl,msglen               ;get string length
        mov     bh,0

        mov     msg[bx],'$'             ;terminate string

        mov     ah,09h                  ;display it
```

```
        mov     dx,offset msg
        int     21h

        mov     ax,4C00h            ;bye-bye
        int     21h
cseg    ends
        end     start
```

**โปรแกรมที่ 8.11 โปรแกรมรับข้อความและแสดงข้อความนั้นกลับมา**

**สรุป**

การประกาศข้อมูลหรือตัวแปรในโปรแกรมภาษาแอสเซมบลีนั้น ทำได้โดยประกาศจองเนื้อที่ในหน่วยความจำในเซกเมนต์ข้อมูล แล้วตั้งเลเบลของข้อมูลนั้นไว้ ในการอ้างถึงข้อมูลในหน่วยความจำตำแหน่งนั้น เราสามารถอ้างโดยใช้เลเบลที่เราประกาศไว้ได้ ดังนั้นการประกาศตัวแปรหรือข้อมูลนั้นจะมีลักษณะเช่นเดียวกับการประกาศเลเบลนั่นเอง ส่วนในการอ้างใช้ข้อมูลหรือตัวแปรที่ประกาศไว้ เราสามารถอ้างโดยใช้ชื่อของเลเบลที่ประกาศไว้ได้ Assembler จะจัดการนำตำแหน่งของข้อมูลนั้นมาแทนค่าให้โดยอัตโนมัติ เรายังสามารถอ้างค่าในหน่วยความจำโดยอ้างสัมพัทธ์กับเลเบลที่เรากำหนดขึ้นได้

การประกาศข้อมูลสำหรับการใช้บริการของ DOS หมายเลข 09h และ 0Ah ฟังก์ชั่นหมายเลข 09h และ 0Ah ของ DOS เป็นฟังก์ชั่นที่ต้องมีการส่งแอดเดรสของข้อมูลในหน่วยความจำ การประกาศข้อมูลสำหรับฟังก์ชั่นหมายเลข 09h จะไม่มีความซับซ้อนมากนัก แต่สำหรับฟังก์ชั่นหมายเลข 0Ah การประกาศข้อมูลที่เหมาะสมจะทำให้เราเขียนโปรแกรมได้ง่ายมากขึ้น การใช้บริการของ DOS ฟังก์ชั่นหมายเลข 0Ah จะเกี่ยวข้องกับการอ่านข้อความส่วนฟังก์ชั่นหมายเลข 09h จะเกี่ยวข้องกับการพิมพ์ข้อความ

**คำถามทบทวน**

1. จงอธิบายความหมายของคำสั่งเทียมสำหรับการระบุขนาดข้อมูลในการจองหน่วยความจำต่อไปนี้
   DB, DW, DD, DQ, และ DT
2. จงแสดงวิธีการจัดเรียงข้อมูลในหน่วยความจำจากการประกาศในส่วนของโปรแกรมที่แสดงอยู่ข้างล่างนี้

```
dseg    segment
data1   dw      1,2
data2   dw      1,2
data3   db      'Cs',14,15
data4   dd      3456h
dseg    ends
```

2. คำสั่งเทียม dup มีไว้เพื่ออะไร
3. อักขระหมายเลข 10 (Line feed) และ 13 (Carriage Return) หมายถึงอะไร

4. จงอธิบายการประกาศข้อมูลแต่ละบรรทัดสำหรับการเรียกใช้ฟังก์ชั่นจากส่วนของโปรแกรมที่แสดงอยู่ข้างล่างนี้

```
dseg    segment
maxlen  db      60
msglen  db      ?
msg     db      40 dup (?)
dseg    ends
```

5. ถ้าเราต้องการพิมพ์ข้อความ “Computer Science SDU” เราสามารถประกาศข้อมูลในหน่วยความจำนี้ได้อย่างไร

6. ข้อความที่สั่งพิมพ์ด้วยบริการหมายเลข 09h จะต้องจบด้วยอักขระใดเสมอ