คำสั่งกระโดดและคำสั่งวนรอบ

- คำสั่งกระโดด
  - คำสั่งกระโดดแบบไม่มีเงื่อนไข
  - คำสั่งกระโดดแบบมีเงื่อนไข
- คำสั่งวนรอบ
- ตัวอย่าง

สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

คำสั่งกระโดด

- แบ่งออกเป็น 2 กลุ่ม
  - คำสั่งกระโดดแบบไม่มีเงื่อนไข : JMP
  - คำสั่งกระโดดแบบมีเงื่อนไข : Jxx
    - ตามแฟล็ก
    - [illegible] JCXZ
- รูปแบบ

J [illegible]

ex

```
        mov   ah,02
        mov   dl,' '
printloop:
        int   21h
        inc   dl
        cmp   dl,128
```

[illegible]

สังเกตว่าคำสั่งกระโดดที่ใช้เงื่อนไขจากค่าของแฟล็ก มักจะใช้ประกอบกับคำสั่ง **CMP**.

สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

คำสั่งกระโดด

คำสั่ง	ความหมาย	เงื่อนไข
คำสั่งกระโดดแบบไม่มีเงื่อนไข		
JMP	Jump	
เงื่อนไขตามแฟล็ก		
แฟล็กศูนย์		
JZ	Jump if Zero	ZF=1
JNZ	Jump if Not Zero	ZF=0
แฟล็กโอเวอร์โฟล์		
JO	Jump if Overflow	OF=1
JNO	J. if Not Overflow	OF=0
แฟล็กทด		
JC	Jump if Carry	CF=1
JNC	Jump if No Carry	CF=0
แฟล็กเครื่องหมาย		
JS	Jump if Sign	SF=1
JNS	Jump if No Sign	SF=0

คำสั่งกระโดด (ต่อ)

คำสั่ง	ความหมาย	เงื่อนไข
เงื่อนไขตามแฟล็ก		
พาริตี้แฟล็ก		
JPO	Jump if Parity Odd	PF=0
JPE	Jump if Parity Even	PF=1
ตัวเลขแบบไม่คิดเครื่องหมาย		
JA	Jump if Above	(CF and ZF)=0
JB	Jump if Below	CF=1
JAE	J. if Above or Equal	CF=0
JBE	J. if Below or Equal	(CF or ZF)=1
ตัวเลขแบบคิดเครื่องหมาย		
JG	Jump if Greater	ZF=0 and SF=OF
JL	Jump if Less	SF<>OF
JGE	J. if Greater or Equal	SF=OF
JLE	J. if Less or Equal	(ZF=1) or (SF<>OF)
เงื่อนไขตามค่าในรีจิสเตอร์		
JCXZ	Jump if CX=0	CX=0

สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

ชื่ออื่น ๆ ของคำสั่งกระโดด

คำสั่ง	ชื่ออื่น	ความหมาย
JZ	JE	Equal
JNZ	JNE	Not Equal
JA	JNBE	Not Below or Equal
JB	JNAE	Not Above or Equal
JAE	JNB	Not Below
JBE	JNA	Not Above
JG	JNLE	Not Less or Equal
JL	JNGE	Not Greater or Equal
JGE	JNL	Not Less
JLE	JNG	Not Greater

สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

ตัวอย่างการใช้คำสั่งกระโดด

EX#1

```
        cmp    ah,10      ;เปรียบเทียบ ah กับ 10
        jz     lab1       ;ถ้าเท่ากันให้กระโดดไปที่ lab1
        mov    bx,2
lab1:   add    cx,10
```

EX#2

```
        cmp    ah,10      ;เปรียบเทียบ ah กับ 10 ถ้ามากกว่า
        jge    tenup      ;หรือเท่ากับให้กระโดดไปที่ lab1
        add    dl,'0'     ;ปรับค่า dl
        jmp    endif      ;กระโดดไปที่ endif
tenup:
        add    dl,'A'     ;ปรับค่า dl
endif:
```

EX#3

```
getonechar:
        mov    ah,1       ; ใช้บริการหมายเลข 1
        int    21h        ; อ่านอักขระ
        cmp    al,'Q'     ; ถ้าไม่เท่ากับ 'Q'
        jne    getonechar ; -> ไปอ่านใหม่
```


สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

ตัวอย่างการใช้คำสั่งกระโดด (2)

ตัวอย่าง

```
        mov     ah,02           ; ใช้บริการหมายเลข 2
        mov     dl,32           ; เริ่มที่ ' '
printloop:
        cmp     dl,128          ; ถ้ามากกว่า ASCII 128
        ja      finish          ; -> จบการทำงาน
        int     21h
        inc     dl
        jmp     printloop       ; พิมพ์ต่อ
finish:
```


สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

คำสั่งวนรอบ

- คำสั่งวนรอบเป็นคำสั่งที่ใช้ในการกระทำซ้ำ โดยใช้รีจิสเตอร์ CX (Counter Register) ในการนับจำนวนครั้งของการกระทำซ้ำ.
- คำสั่งในกลุ่มนี้คือ
  - **LOOP** : คำสั่งที่พิจารณาค่าของ CX อย่างเดียว
  - **LOOPZ, LOOPNZ** : พิจารณาแฟล็กร่วมด้วย
- รูปแบบ

  LOOP ***label***
- การทำงานของคำสั่ง LOOP
  - ลดค่าของ CX ลงหนึ่ง **โดยไม่กระทบแฟล็ก**
  - ถ้า CX ยังมีค่ามากกว่าศูนย์ กระโดดไปทำงานที่เลเบลที่ระบุ
- คำสั่ง LOOP มีการทำงานเทียบเท่ากับ

  DEC **CX**

  JNZ ***label***

แต่ไม่มีการกระทบแฟล็ก

ตัวอย่างการใช้คำสั่งวนรอบ

ex

```
        mov     cx,20           ; ทำซ้ำ 20 ครั้ง
        mov     bl,1            ; เริ่มที่ 1
        mov     dx,0            ; ค่าเริ่มต้น = 0
addnumber:
        add     dl,bl           ; บวก 8 บิตล่าง
        adc     dh,0            ; บวกตัวทด
        inc     bl
        loop    addnumber       ; ทำซ้ำ
```

- คำสั่ง JCXZ

ในกรณีที่รีจิสเตอร์ CX มีค่าเท่ากับศูนย์ก่อนการทำงานของคำสั่ง LOOP. ค่าของ CX จะถูกปรับค่าเป็น 0FFFFh และการทำงานจะผิดพลาด. เราสามารถใช้คำสั่ง JCXZ ในการป้องกันความผิดพลาดกรณีนี้ได้ ดังตัวอย่าง.

```
        initialize
        jcxz endloop
label1:
        actions
        loop label1
endloop:
```

คำสั่ง LOOPZ และ LOOPNZ

- คำสั่ง LOOPZ และ LOOPNZ มีการทำงานเหมือนกับคำสั่ง LOOP แต่จะนำค่าของแฟล็กศูนย์มาใช้ในการพิจารณาการกระโดดด้วย.
  - LOOPZ จะกระโดดกลับไปทำงานถ้าค่าของ CX ที่ลดแล้วมีค่าไม่เท่ากับศูนย์ และค่าของแฟล็กศูนย์มีค่าเป็นหนึ่ง (Zero.)

    ➡ (CX <> 0) and (ZF = 1)
  - LOOPNZ จะกระโดดกลับไปทำงานถ้าค่าของ CX ที่ลดแล้วมีค่าไม่เท่ากับศูนย์ และค่าของแฟล็กศูนย์มีค่าเป็นศูนย์ (Not Zero.)

    ➡ (CX <> 0) and (ZF = 0)

ตัวอย่างการใช้คำสั่ง LOOPNZ

ex

```
.data
datalist  dw      100 dup(?)

.code
            …
           mov    bx,offset datalist
           mov    cx,100
                             ;***
           sub    bx,2       ;for inc
                             ;***
checkdata:
           add    bx,2
           cmp    dx,[bx]
           loopnz checkdata
           jz     found

           ; not found
            …
found:
           ; found
            …
```

โปรแกรมนี้จะค้นหาข้อมูลใน datalist ที่มีค่าเท่ากับค่าในรีจิสเตอร์ DX. สังเกตว่าเราลดค่าของ BX ก่อนที่จะเข้าไปทำงานในวงรอบ เพื่อป้องกันปัญหาในการตรวจสอบข้อมูลตัวแรก.

สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

ตัวอย่างการใช้คำสั่ง LOOPZ

ex

```
        cmp     byte ptr [BX],' '
        jnz     found
        mov     cx,100
findnotspace:
        inc     bx
        cmp     byte ptr [bx],' '
        loopz findnotspace
        jnz     found

        ; not found
         …
found:
        ; found
         …
```

สังเกตว่าในกรณีนี้เราไม่ได้ลดค่าของ BX แต่เรานำข้อมูลตัวแรกมาตรวจสอบเสียก่อน. เราสามารถเขียนโปรแกรมให้มีโครงสร้างได้หลายรูปแบบ. ถ้าเราใช้วิธีพิเศษดังตัวอย่างที่แล้ว เราควรจะต้องใส่หมายเหตุให้ชัดเจน.

สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

ตัวอย่างโปรแกรม

- โปรแกรมรับการกดปุ่มจากผู้ใช้แล้วแสดงค่ารหัสแอสกีของอักขระนั้นเป็นตัวเลขฐานสิบหก.
  - ปัญหา
    - รหัสแอสกีของอักษรที่จะนำมาแสดงมีสองชุด
      - '0' - '9' มีรหัสแอสกีเป็น 48 - 57
      - 'A' - 'F' มีรหัสแอสกีเป็น 65 - 69

ex

```
        mov   dl,al
        cmp   al,9
        ja    nineup1
        add   dl,'0'
        jmp   disp1
nineup1:
        add   dl,'A'-10
disp1:
        mov   ah,02
        int   21h
```


สัณฑิติ พัชรรุ่งเรือง
จิตร์ทัศน์ ฝักเจริญผล

ตัวอย่างโปรแกรม

โปรแกรมแสดงข้อความกลับหน้าหลัง

- รับข้อความจากผู้ใช้แล้วแสดงข้อความแบบกลับหน้าหลัง.
  - รับข้อความ : Function 0Ah
  - แสดงตัวอักษร : Function 02h

ex

```
.data
maxlen   db    30
strlen   db    ?
str      db    30 dup (?)

.code
          …
         mov   cl,strlen
         mov   ch,0
         mov   bx,offset str
         add   bx,cx
         dec   bx            ;last char
         mov   ah,02
printchar:
         mov   dl,[bx]
         int   21h
         dec   bx
         loop  printchar
          …
```