สัปดาห์ที่ 6

คำสั่งโอนย้ายข้อมูล แฟล็กและคำสั่งคณิตศาสตร์

Debug Programming Training

เริ่มจากคำว่าบั๊ก (Bug) ซึ่งในระบบคอมพิวเตอร์หมายถึงข้อผิดพลาดที่เกิดขึ้นหรือทำงานของโปรแกรมไม่ตรงกับความต้องการ และคำว่าดีบั๊ก (Debug) ก็จะหมายถึงการเข้าไปตรวจสอบดูการทำงานทีละคำสั่งจนพบและแก้ไขข้อผิดพลาดนั้นให้สามารถทำงานตามความต้องการได้ โดยมีที่มาของคำนี้มาจาก ยุคแรกๆของคอมพิวเตอร์ ซึ่งเครื่อง Mark-I ได้เกิดขัดข้องขึ้นมา โดยหาสาเหตุไม่พบ ทำให้ช่างเทคนิคต้องใช้เวลาอยู่นานเพื่อค้นหาจุดบกพร่องในเครื่องฯ จนกระทั่งได้ไปพบแมลง (Bug) ตัวเล็กๆ ติดอยู่บริเวณจุดสัมผัสของหลอดสูญญากาศ และเมื่อเอาแมลงตัวนี้ออกไป (Debug) ทำให้เครื่อง Mark-I ก็สามารถทำงานต่อไปได้ปกติ ดังนั้นการใช้โปรแกรมดีบั๊ก ก็จะเป็นการเข้าไปตรวจสอบส่วนต่างๆ ของโปรแกรม และสามารถเขียนโปรแกรมในภาษาแอสเซมบลีสั้นๆได้

โปรแกรมดีบั๊กในระบบปฏิบัติการดอส (DOS) จะมีไฟล์ชื่อ DEBUG.COM ส่วนผู้ที่ใช้ระบบปฏิบัติการวินโดวส์9X จะมีไฟล์ชื่อ DEBUG.EXE ซึ่งจะเก็บอยู่ในไดเร็กเทอรีชื่อ C:\WINDOWS\COMMAND

การเรียกใช้โปรแกรม

1. เมื่ออยู่ในดอส จะสามารถเรียกใช้ดีบั๊กได้โดยพิมพ์เพียงชื่อไฟล์คือ debug ดังนี้

C:\>debug

2. จะเข้าสู่โปรแกรมดีบั๊ก โดยจะแสดงเครื่องหมาย Prompt ของดีบั๊กคือเครื่องหมาย – ขึ้นมา เพื่อแสดงการเตรียมพร้อมรับคำสั่ง

C:\>debug

รายการคำสั่งของดีบั๊ก

เราสามารถเรียกดูรายการคำสั่งจะต้องพิมพ์เครื่องหมาย ? ซึ่งหมายถึง Help

-?

จะปรากฏรายการคำสั่งดังนี้

ความหมาย	คำสั่ง
assemble	A [address]
compare	C range address
dump	D [range]
enter	E address [list]
fill	F range list
go	G [=address] [addresses]
hex	H value1 value2
input	I port
load	L [address] [drive] [firstsector] [number]
move	M range address
name	N [pathname] [arglist]
output	O port byte
proceed	P [=address] [number]
quit	Q
register	R [register]
search	S range list
trace	T [=address] [value]
unassemble	U [range]
write	W [address] [drive] [firstsector] [number]
allocate expanded memory	XA [#pages]
deallocate expanded memory	XD [handle]
map expanded memory pages	XM [Lpage] [Ppage] [handle]
display expanded memory status	XS

คำสั่ง H

เป็นคำสั่ง Hexarithmetic ใช้ในการบวกหรือลบเลขฐานสิบหก 2 จานวน โดยจะเขียนตามหลังคำสั่ง H เช่น

-H 4 2

0006 0002

-

จะเป็นทั้งการบวกและการลบเลข 4 กับเลข 2 ในฐานสิบหก นั่นคือคำตอบที่ได้คือ 0006 คือผลบวก และ 0002 คือผลจากการลบ 4 ด้วยเลข 2
ถ้าลองใช้คำสั่งต่อไปนี้

-H 2 4

0006 FFFE

-

จะพบว่ามีสิ่งที่ไม่ปกติเกิดขึ้นที่ผลของการลบ กล่าวคือ 2-4 จะต้องได้ผลลัพธ์เป็น -2 แต่กลับออกมาเป็น FFFE ซึ่งตามความเป็นจริงแล้วเป็นคาตอบที่ถูกต้องเพราะ -2 ถ้าแปลงเป็นเลขฐานสิบหกคือ ให้ 2 แปลงเป็นฐานสองจะได้ 0000 0000 0000 0010 ถ้าติดลบจะต้องทาให้อยู่ในรูป 2's คอมพลีเมนต์ คือ

1's ของ 0000 0000 0000 0010 คือ 1111 1111 1111 1101

1111 1111 1111 1101

บวก 1 : 1

2's : 1111 1111 1111 1110

เมื่อแปลง 1111 1111 1111 1110 เป็นเลขฐานสิบหกจะได้ F F F E => FFFE นั่นเอง
ทำให้ผลลัพธ์ของการลบ 2-4 ได้ออกมาเป็น FFFE

เนื่องจากหน่วยในการเก็บข้อมูลเป็นแบบเวิร์ด(WORD) ดังนั้นจะพบว่ามีการใช้ตัวเลขฐานสิบหกเพียง 4 หลักเท่านั้น ถ้าใช้เกินกว่านี้ จะเกิดข้อผิดพลาดเช่น

-H A0000 1000

^ Error

-

ถ้าผลลัพธ์ออกมาเกิน 4 หลัก ก็จะมีการตัดส่วนที่เกินออกไป จะรับเพียง 4 หลักขวาสุดเท่านั้น เช่น

A000+D000 น่าจะได้ 17000 แต่จะได้ผลลัพธ์เพียง 7000 ส่วน 1 ถูกตัดทิ้งไป ส่วน A000-D000 ก็
น่าจะได้คำตอบคือ FFFFD000 แต่จะได้ผลลัพธ์เพียง D000 ส่วน FFFF ถูกตัดทิ้งไป ดังนั้นไม่ควรใช้ตัวเลขที่เกินกว่า 4 หลัก

-H A000 D000

7000 D000

-

คำสั่ง D

มาจากคำว่า DUMP จะเป็นการแสดงค่าในหน่วยความจำตามตำแหน่งที่ระบุตามหลังคำสั่ง D เพื่อเป็นการตรวจสอบความถูกต้องของข้อมูลในหน่วยความจำ โดยจะแสดงตำแหน่งในหน่วยความจำโดยมีรูปแบบคือ ค่าเซกเมนต์(Segment) : ค่าออฟเซต (Offset) ซึ่งเป็น Logical address จะหาได้จากค่าที่เก็บใน CS และ IP (ถ้าใช้คำสั่ง R) สำหรับการหาตำแหน่งบนหน่วยความจำจริงๆ (Physical Address) ก็จะสามารถหาค่าได้โดย

ค่าเซกเมนต์ x 10h + ค่าออฟเซต

และถัดมาจะแสดงข้อมูลในรูปตัวเลขฐานสิบหกชุดละ 2 ตัว ขนาด 1 ไบต์ ในแต่ละบรรทัดจะมีอยู่ 16 ไบต์ โดยจะมีตำแหน่งของไบต์เริ่มต้นด้านซ้ายสุดคือ ตำแหน่ง 0 จนถึงขวาสุดคือตำแหน่ง F และรหัสแอสกีด้านหลังสุด

```
-D 100
ตำแหน่งที่   0 1  2  3  4  5  6  7  8  9 A  B  C  D E  F
10A6:0100 51 52 55 8B 36 CD E2 56-33 ED AC 0A C0 74 03 45 QRU.6..V3....t.E
10A6:0110 EB F8 5E 33 C9 8B D1 56-AC 3C 2E 75 34 00 95 10 ..^3...V.<.u4...
10A6:0120 3C 2A 75 03 83 CA 02 3C-3F 75 03 83 CA 04 0A C0 <*u....<?u......
10A6:0130 74 03 41 EB E3 5E E3 0B-F7 C2 01 00 74 1C 80 3C t.A..^......t..<
10A6:0140 2E 74 47 83 3E D3 E2 02-75 0A 80 3E DA E3 3A 75 .tG.>...u..>..:u
10A6:0150 03 E8 34 FF 5D 5A 59 5F-5E C3 83 3E D3 E2 02 75 ..4.]ZY_^..>...u
10A6:0160 0B 80 7C 01 3A 75 05 E8-1E FF EB E8 03 F5 80 3E ..|.:u.........>
10A6:0170 D1 E2 00 74 06 F7 C2 06-00 75 D9 C7 04 2E 2A C6 ...t.....u....*.
-
```

*** ถ้าไม่ระบุตำแหน่ง (Address) หลังคำสั่ง D ก็จะเป็นการแสดงหน่วยความจำปัจจุบันที่กำลังทำงานอยู่

```
-D
ตำแหน่งที่   0  1  2  3  4 5  6 7  8  9 A  B  C D  E  F
10A6:0880 8C DE 89 77 09 8B F7 8B-3E 86 DE 03 F9 3B 3E 82 ...w....>....;>.
10A6:0890 DE 7D 15 2B F9 FC F3 A4-B0 00 AA 89 3E 86 DE 9D .}.+........>...
```

คำสั่ง F

มาจากคำว่า Fill เป็นคำสั่งให้เอาค่าคงที่ ไปใส่ในหน่วยความจำตามหมายเลขที่กำหนดตามหลังคำสั่ง F เช่น ต้องการนำเอาค่า 61 มาใส่ในหน่วยความจำตั้งแต่ตำแหน่ง 100 ถึง 10F

เริ่มจาก dump ตรวจสอบดูหน่วยความจำตำแหน่งเริ่มต้น 100

-d 100

10A6:0100 60 60 60 60 60 60 60 60-00 00 00 00 00 00 00 00 ````````........

10A6:0110 65 65 65 33 C9 8B D1 56-AC 3C 2E 75 34 00 95 10 eee3...V.<.u4...

10A6:0120 3C 2A 75 03 83 CA 02 3C-3F 75 03 83 CA 04 0A C0 <*u....<?u.....

10A6:0130 74 03 41 EB E3 5E E3 0B-F7 C2 01 00 74 1C 80 3C t.A..^.....t..<

10A6:0140 2E 74 47 83 3E D3 E2 02-75 0A 80 3E DA E3 3A 75 .tG.>...u..>..:u

10A6:0150 03 E8 34 FF 5D 5A 59 5F-5E C3 83 3E D3 E2 02 75 ..4.]ZY_^..>...u

10A6:0160 0B 80 7C 01 3A 75 05 E8-1E FF EB E8 03 F5 80 3E ..|.:u.........>

10A6:0170 D1 E2 00 74 06 F7 C2 06-00 75 D9 C7 04 2E 2A C6 ...t.....u....*.

เติมค่า 61 ซึ่งมีค่าหมายถึงอักษร a ลงไปในตำแหน่ง 100 ถึง 10F

-F 100 10F 61

ใช้คำสั่ง D ตรวจสอบดูความเปลี่ยนแปลงในหน่วยความจำตำแหน่งเริ่มต้น 100

-D 100

10A6:0100 61 61 61 61 61 61 61 61-61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaa

10A6:0110 65 65 65 33 C9 8B D1 56-AC 3C 2E 75 34 00 95 10 eee3...V.<.u4...

10A6:0120 3C 2A 75 03 83 CA 02 3C-3F 75 03 83 CA 04 0A C0 <*u....<?u.....

10A6:0130 74 03 41 EB E3 5E E3 0B-F7 C2 01 00 74 1C 80 3C t.A..^.....t..<

10A6:0140 2E 74 47 83 3E D3 E2 02-75 0A 80 3E DA E3 3A 75 .tG.>...u..>..:u

10A6:0150 03 E8 34 FF 5D 5A 59 5F-5E C3 83 3E D3 E2 02 75 ..4.]ZY_^..>...u

10A6:0160 0B 80 7C 01 3A 75 05 E8-1E FF EB E8 03 F5 80 3E ..|.:u.........>

10A6:0170 D1 E2 00 74 06 F7 C2 06-00 75 D9 C7 04 2E 2A C6 ...t.....u....*.

-

คำสั่ง R

มาจากคำว่า Register ซึ่งเป็นการแสดงค่าที่อยู่ในรีจิสเตอร์ทั้งหมดออกมา หรือสามารถใช้เปลี่ยนแปลงค่าในรีจิสเตอร์ก็ได้

-R

AX=0000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000

DS=10A6 ES=10A6 SS=10A6 CS=10A6 IP=0100 NV UP EI PL NZ NA PO NC

10A6:0100 61 DB 61

จากตัวอย่างข้างต้นจะสามารถมองเห็นค่าของริสเตอร์ต่างๆได้ เช่น รีจิสเตอร์ AX มีค่าเป็น 0000 และค่าของ BX CX จนถึง IP ตามลาดับ โดยค่าที่อยู่ในรีจิสเตอร์แต่ละตัวยังอยู่ในรูปเลขฐานสิบหกเสมอ ถัดมาจะแสดงแฟล็กรีจิสเตอร์ (Flags Register) ส่วนในบรรทัดที่สาม จะแสดงตำแหน่งที่อยู่ของคำสั่งใน ซึ่งอยู่ในรูปรหัสนีมอนิก (Mnemonic) รีจิสเตอร์ก็เหมือนตัวแปรในภาษาซีหรือปาสคาล แต่รีจิสเตอร์ไม่สามารถเพิ่มจานวนได้ ดังนั้นเราสามารถเปลี่ยนค่ารีจิสเตอร์ AX ให้เป็น 100 ได้คือ

```
-R AX
AX 0000
:0100
-R
AX=0100 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=10A6 ES=10A6 SS=10A6 CS=10A6 IP=0100 NV UP EI PL NZ NA PO NC
10A6:0100 61 DB 61                    Flags Register
```

ค่าใน AX ถูกเปลี่ยนไป

ความหมายของแฟล็กรีจิสเตอร์

ชื่อแฟล็ก	Set(1)	Clear (0)
Overflow(yes/no)	OV	NV
Direction(decrement/increment)	DN	UP
Interrupt(enable/disable)	EI	DI
Sign(negative/positive)	NG	PL
Zero(yes/no)	ZR	NZ
Auxiliary carry (yes/no)	AC	NA
Parity(even/odd)	PE	PO
Carry(yes/no)	CY	NC

คำสั่ง E

มาจากคำว่า Enter ซึ่งจะเป็นการป้อนค่าคงที่ลงในหน่วยความจำ ตามตำแหน่งที่ระบุ เช่น ต้องการป้อนค่า 61 (อักษร a) ลงในหน่วยความจำตำแหน่งที่ 100

เริ่มจากตรวจสอบหน่วยความจำในตำแหน่งเริ่มต้นที่ 100

-D 100

10A6:0100 61 61 61 61 61 61 61 61-61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaa

10A6:0110 65 65 65 33 C9 8B D1 56-AC 3C 2E 75 34 00 95 10 eee3...V.<.u4...

10A6:0120 3C 2A 75 03 83 CA 02 3C-3F 75 03 83 CA 04 0A C0 <*u....<?u.....

10A6:0130 74 03 41 EB E3 5E E3 0B-F7 C2 01 00 74 1C 80 3C t.A..^.....t..<

10A6:0140 2E 74 47 83 3E D3 E2 02-75 0A 80 3E DA E3 3A 75 .tG.>...u..>..:u

10A6:0150 03 E8 34 FF 5D 5A 59 5F-5E C3 83 3E D3 E2 02 75 ..4.]ZY_^..>...u

10A6:0160 0B 80 7C 01 3A 75 05 E8-1E FF EB E8 03 F5 80 3E ..|.:u.....>

10A6:0170 D1 E2 00 74 06 F7 C2 06-00 75 D9 C7 04 2E 2A C6 ...t.....u....*.

ป้อนค่า 62 ลงในตำแหน่งที่ 100

-E 100

10A6:0100 61.62

**** สามารถกำหนดตำแหน่งใดๆ ในหน่วยความจำได้ โดยใช้ค่าออฟเซตของแต่ละบรรทัด+ตำแหน่งของข้อมูลของแต่ละบรรทัด (0 ถึง F) เช่น 130+ 5 ก็จะใช้ E 135 เป็นต้น เมื่อตรวจสอบการเปลี่ยนแปลงจะพบว่า ค่า 61 (ตัวอักษร a) ถูกเปลี่ยนไปเป็น 62 (ตัวอักษร b)

-D 100

10A6:0100 62 61 61 61 61 61 61 61-61 61 61 61 61 61 61 61 baaaaaaaaaaaaaaa

10A6:0110 65 65 65 33 C9 8B D1 56-AC 3C 2E 75 34 00 95 10 eee3...V.<.u4...

10A6:0120 3C 2A 75 03 83 CA 02 3C-3F 75 03 83 CA 04 0A C0 <*u....<?u.....

10A6:0130 74 03 41 EB E3 5E E3 0B-F7 C2 01 00 74 1C 80 3C t.A..^.....t..<

10A6:0140 2E 74 47 83 3E D3 E2 02-75 0A 80 3E DA E3 3A 75 .tG.>...u..>..:u

10A6:0150 03 E8 34 FF 5D 5A 59 5F-5E C3 83 3E D3 E2 02 75 ..4.]ZY_^..>...u

10A6:0160 0B 80 7C 01 3A 75 05 E8-1E FF EB E8 03 F5 80 3E ..|.:u.....>

10A6:0170 D1 E2 00 74 06 F7 C2 06-00 75 D9 C7 04 2E 2A C6 ...t.....u....*.

-

คำสั่ง A

มาจากคาว่า Assemble ใช้สำหรับเขียนโปรแกรมภาษาแอสเซมบลี ในดีบั๊ก ซึ่งมีข้อกำหนดว่าจะต้องเริ่มต้นที่ตำแหน่งออฟเซตที่ 100 เสมอ เช่น

```
-A 100
10A6:0100 MOV DL,61
10A6:0102 MOV AH,2
10A6:0104 INT 21
10A6:0106 INT 20
10A6:0108
```

คำสั่ง T

มาจากคำว่า Trace ใช้สำหรับแสดงการทำงานของโปรแกรมทีละคำสั่ง เช่นต้องการตรวจสอบการทางานของโปรแกรม MEM.EXE จะเริ่มต้นที่ debug mem.exe แล้วจึง ติดตามดูคำสั่งทีละขั้นตอนโดยละเอียด

```
C:\WIN98\COMMAND>DEBUG MEM.EXE
-T
AX=0000 BX=0000 CX=7B92 DX=0000 SP=0080 BP=0000 SI=0000 DI=0000
DS=10CB ES=10CB SS=19A8 CS=1777 IP=0012 NV UP EI PL NZ NA PO NC
1777:0012 8CC0 MOV AX,ES
-T
AX=10CB BX=0000 CX=7B92 DX=0000 SP=0080 BP=0000 SI=0000 DI=0000
DS=10CB ES=10CB SS=19A8 CS=1777 IP=0014 NV UP EI PL NZ NA PO NC
1777:0014 051000 ADD AX,0010
-T
AX=10DB BX=0000 CX=7B92 DX=0000 SP=0080 BP=0000 SI=0000 DI=0000
DS=10CB ES=10CB SS=19A8 CS=1777 IP=0017 NV UP EI PL NZ NA PE NC
1777:0017 0E PUSH CS
-T
```

คำสั่ง G

มาจากคาว่า Go ใช้สำหรับสั่งให้โปรแกรมที่เขียนจากคำสั่ง A ให้ทำงานตามขั้นตอน

```
-A 100
10A6:0100 MOV DL,61
10A6:0102 MOV AH,2
10A6:0104 INT 21
10A6:0106 INT 20
10A6:0108
-G
a
Program terminated normally
-
```

คำสั่ง U

มาจากคำว่า Unassembled ใช้สำหรับแปลภาษาเครื่อง (Machine Language) ให้เป็นภาษาแอสเซมบลี ซึ่งจะทำงานตรงกันข้ามกับคำสั่ง A โดยคำสั่ง U สามารถระบุตำแหน่งเริ่มต้น, ตำแหน่งสุดท้ายของหน่วยความจำ ที่ต้องการแสดงคำสั่ง

```
-U 100,10A
10A6:0100 51          PUSH CX
10A6:0101 52          PUSH DX
10A6:0102 55          PUSH BP
10A6:0103 8B36CDE2    MOV SI,[E2CD]
10A6:0107 56          PUSH SI
10A6:0108 33ED        XOR BP,BP
10A6:010A AC          LODSB
```

ภาษาเครื่อง ภาษาแอสเซมบลี

คำสั่ง N

มาจากคำว่า Name ใช้สำหรับตั้งชื่อไฟล์ ในการบันทึกโปรแกรมที่เขียนในดีบั๊ก โดยปกติโปรแกรมที่เขียนในดีบั๊กจะมีขนาดไม่เกิน 64K ดังนั้นจะเป็นไฟล์ที่มีนามสกุลเป็น .com

```
- N printa.com
```

คำสั่ง W

มาจากคำว่า Write ใช้สำหรับบันทึกไฟล์ลงไว้ในดิสก์ โดยไฟล์นั้นจะผ่านการตั้งชื่อด้วยคำสั่ง N มาก่อนแล้ว

คำสั่ง Q

มาจากคำว่า Quit ใช้สำหรับออกจากโปรแกรมดีบั๊ก

-Q

หรือ

-q

ตัวอย่าง ขั้นตอนการเขียนโปรแกรมภาษาแอสเซมบลี ในดีบั๊ก

การแสดงตัวอักษร a ออกทางจอภาพ

ขั้นตอน

1. เข้าสู่ debug

2. ป้อนรหัสนีมอนิค ด้วยคำสั่ง A ณ ตำแหน่งที่ 100

-A 100

10A6:0100 MOV DL,61

10A6:0102 MOV AH,02

10A6:0104 INT 21

10A6:0106 INT 20

10A6:0108

*** MOV AH,02 และ INT 21 เป็นฟังก์ชันในการแสดงตัวอักษรออกทางจอภาพ จะต้องใช้คู่กัน ส่วน INT 20 เป็น Interrupt ที่หมายถึงจบโปรแกรมและออกไปสู่ดอส

3. ให้ใช้คาสั่ง G เพื่อทดสอบโปรแกรม จะได้เป็น

-G

a

Program terminated normally

-

4. เมื่อได้ผลลัพธ์ตามที่ต้องการแล้ว ก็จะต้องตั้งชื่อไฟล์ โดยใช้คำสั่ง N

-N printa.com

-

5. หลังจากตั้งชื่อเสร็จแล้ว ก็จะเป็นการกำหนดขนาดของโปรแกรม ซึ่งจะจัดเก็บขนาดของโปรแกรมไว้ที่รีจิสเตอร์ BX และ CX ถ้าโปรแกรม มีขนาดน้อยกว่าหรือเท่ากับ 64K จะใช้เฉพาะ CX ส่วน BX จะเป็น 0 แต่ถ้าโปรแกรมมีขนาดเกิน 64K จะต้องกำหนดค่าให้ BX ด้วย จากโปรแกรมนี้จะนับขนาดได้ 8 ไบต์ ดังนั้น CX จะมีค่าเป็น 8 ส่วน BX=0 *** ค่า 8 นับจากค่าออฟเซตที่ใช้ คือ

10A6:0100 MOV DL,61

10A6:0102 MOV AH,02

10A6:0104 INT 21

10A6:0106 INT 20

10A6:0108

ดังนั้นจะกำหนดดังนี้

-RBX

BX 0000

:0000

-RCX

CX 0000

:0008

-

6. และบันทึกไฟล์โดยใช้คำสั่ง W

-W

Writing 00008 bytes

-

เป็นอันเสร็จสิ้นการเขียนและบันทึกโปรแกรม ใช้คำสั่ง DIR printa.com เพื่อตรวจสอบ

C:\>dir printa.com

 Volume in drive C has no label

 Volume Serial Number is 1529-10F3

 Directory of C:\

    PRINTA COM 8 09-10-00 9:32p

 1 file(s) 8 bytes

 0 dir(s) 8,003,584 bytes free

C:\>

ตัวอย่าง การพิมพ์ข้อความ Hello ออกทางจอภาพ

ขั้นตอน

1. เข้าสู่โปรแกรมดีบั๊ก

2. ป้อนเลขฐานสิบหกที่เป็นรหัสแอสกีของตัวอักษร ที่ประกอบเป็นข้อความ Hello

H=48 e=65 l=6C l=6C o=6F โดยใช้คำสั่ง E ลงในตำแหน่งที่ 200

-E 200

10A6:0200 40.48

-E 201

10A6:0201 30.65

-E 202

10A6:0202 33.6C

-E 203

10A6:0203 E7.6C

-E 204

10A6:0204 43.6F

- E 205

10A6:0205 01.24

*** จะต้องกำหนด $ ไว้หลังข้อความที่ต้องการแสดงเสมอ เพราะถ้าลืมใส่โปรแกรมจะแสดงข้อมูลในหน่วยความจำทั้งหมด ซึ่ง $ มีค่าเลขฐานสิบหกคือ 24 ถูกใส่โดยคำสั่ง E

3. ใช้คำสั่ง D ที่ตำแหน่ง 200 เพื่อตรวจสอบการใส่ข้อมูล

-D 200

10A6:0200 48 65 6C 6C 6F 24 EB 60-57 06 1E 07 8D 7C 02 81 Hello$.`W....|..

10A6:0210 3C 5C 5C 74 02 47 47 FC-32 C0 B9 FF FF F2 AE F7 <\\t.GG.2.....

10A6:0220 D1 B0 5C 8D 7C 02 F2 AE-BB 00 00 75 19 F2 AE 4F ..\.|.....u...O

10A6:0230 32 C0 26 86 05 50 8B D6-33 DB B8 A0 71 33 C9 F9 2.&..P..3...q3..

10A6:0240 CD 21 58 26 86 05 07 5F-EB 96 2E C7 06 27 E7 00 .!X&..._.....'..

10A6:0250 3B 2E C7 06 29 E7 00 47-2E C7 06 2F E7 00 6C 2E ;...)..G.../..l.

10A6:0260 C7 06 33 E7 00 43 32 C0-5B 59 5A C3 B8 4E 71 2E ..3..C2.[YZ..Nq.

10A6:0270 80 3E D1 E2 00 75 06 E8-6F 00 B8 00 4E 33 F6 CD .>...u..o...N3..

4. ป้อนรหัส ด้วยคำสั่ง A ณ ตำแหน่งที่ 100

-A 100

10A6:0100 MOV DX,0200

10A6:0103 MOV AH,09

10A6:0105 INT 21

10A6:0107 INT 20

10A6:0109

*** MOV AH,09 และ INT 21 เป็นฟังก์ชันในการแสดงข้อความในหน่วยความจำ ที่กำหนดตำแหน่งด้วยค่า DX ให้ออกทางจอภาพ ซึ่งทั้งสองจะต้องใช้คู่กัน

5. ให้ใช้คำสั่ง G เพื่อทดสอบโปรแกรม จะได้เป็น

-G

Hello

Program terminated normally

-

*** โปรแกรมนี้ ถ้าบันทึกไว้ แล้วเรียกใช้งาน จะไม่แสดงคำว่า Hello ออกมาเพราะ ข้อมูลที่อยู่ในหน่วยความจำถูกแทนที่ด้วยค่าอื่นๆไปแล้ว ดังนั้นผลลัพธ์จะออกมาไม่เป็น Hello

งานชิ้นที่ 3

1. ให้ใส่ค่า ข้อมูลต่อไปนี้ลงในหน่วยความจาตำแหน่งเริ่มต้นที่ 200

   01 50 41 59 41 50 03 59 4F 55 02

2. ให้เรียกดูหน่วยความจำแล้วสังเกตผลลัพธ์

3. ให้ใช้คำสั่งในการบวกและลบค่าต่อไปนี้ แล้วสังเกตค่า รีจิสเตอร์ โดยเฉพาะค่าแฟล็ก

   3.1 000F กับ 000D

   3.2 0001 กับ 0004

   3.3 4000 กับ FFFE

   3.4 1111 กับ 1000

4. ใน C:\> ให้ใช้คำสั่ง Debug command.com จากนั้นถ้าต้องการแสดงทั้งภาษาเครื่อง และภาษาแอสเซมบลีของไฟล์ command.com จะใช้คำสั่งอะไร และให้สังเกตผลลัพธ์

5. ให้เขียนโปรแกรมสาหรับแสดงตัวอักษร * ออกทางจอภาพ และให้บันทึกไฟล์ชื่อ star.com

6. ใน C:\> ให้ใช้คำสั่ง Debug star.com แล้วให้ทดสอบโปรแกรมนี้ทีละขั้นตอนพร้อมกับสังเกตรีจิสเตอร์ด้วย *** สังเกตว่าคำสั่งในโปรแกรมนี้จบที่ตำแหน่งใด

ตารางรหัสแอสกีของตัวอักษร

ตารางแสดงรหัส ASCII แทนตัวอักษรภาษาอังกฤษและภาษาไทย

	b7	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	b6	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	b5	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	b4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
b3	b2	b1	b0															
0	0	0	0					0	@	P	'	p				ฐ	ภ	ะ	เ	o
0	0	0	1				!	1	A	Q	a	q			ก	ท	ม	ั	แ	๑
0	0	1	0				"	2	B	R	b	r			ข	ฒ	ย	า	โ	๒
0	0	1	1				#	3	C	S	c	s			ฃ	ณ	ร	ำ	ใ	๓
0	1	0	0				$	4	D	T	d	t			ค	ต	ฤ	ี	ไ	๔
0	1	0	1				%	5	E	U	e	u			ฅ	ถ	ล	ึ	ๆ	๕
0	1	1	0				&	6	F	V	f	v			ฆ	ท	ฦ	ื	็	๖
0	1	1	1				'	7	G	W	g	w			ง	ธ	ว	ุ	่	๗
1	0	0	0				(	8	H	X	h	x			จ	น	ศ	ู	้	๘
1	0	0	1				)	9	I	Y	i	y			ฉ	บ	ษ	ฺ	๊	๙
1	0	1	0				*	:	J	Z	j	z			ช	ป	ส		๋	¶
1	0	1	1				+	;	K	[	k	{			ซ	ผ	ห		์	
1	1	0	0				,	<	L	\	l	|			ฌ	ฝ	ฬ		?	
1	1	0	1				-	=	M	]	m	}			ญ	ฟ	อ			
1	1	1	0				.	>	N	^	n	~			ฎ	พ	ฮ			
1	1	1	1				/	?	O	-	o				ฏ	ฟ	ฯ	฿		