

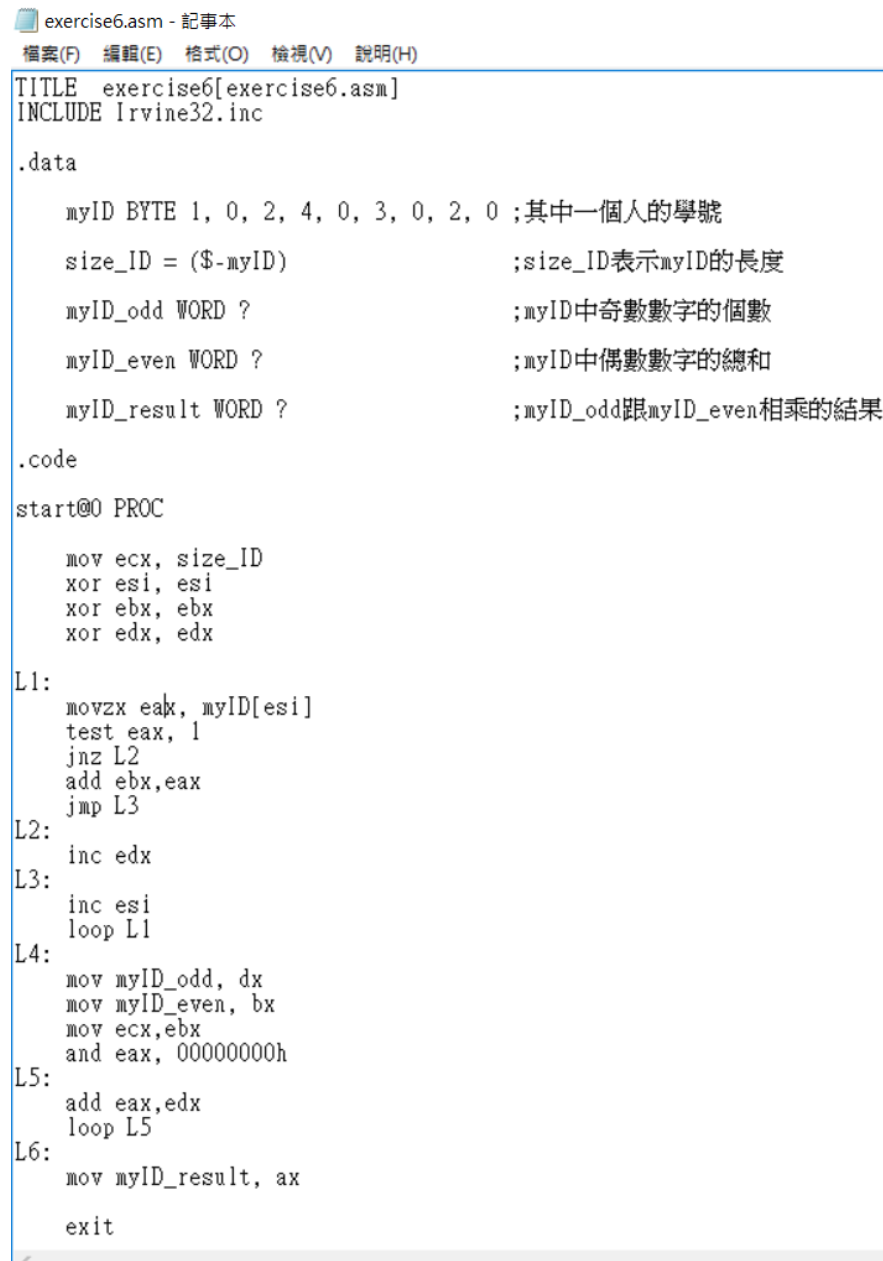
Lab Homework Week 6 Report

Group 66: 102403015 程祥恩、102403016 邱威穎、102403020 曾子軒

Objective:

To understanding how to write simple Assembly program and program structure.

exercise6.asm code:



```
exercise6.asm - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)
TITLE exercise6[exercise6.asm]
INCLUDE Irvine32.inc

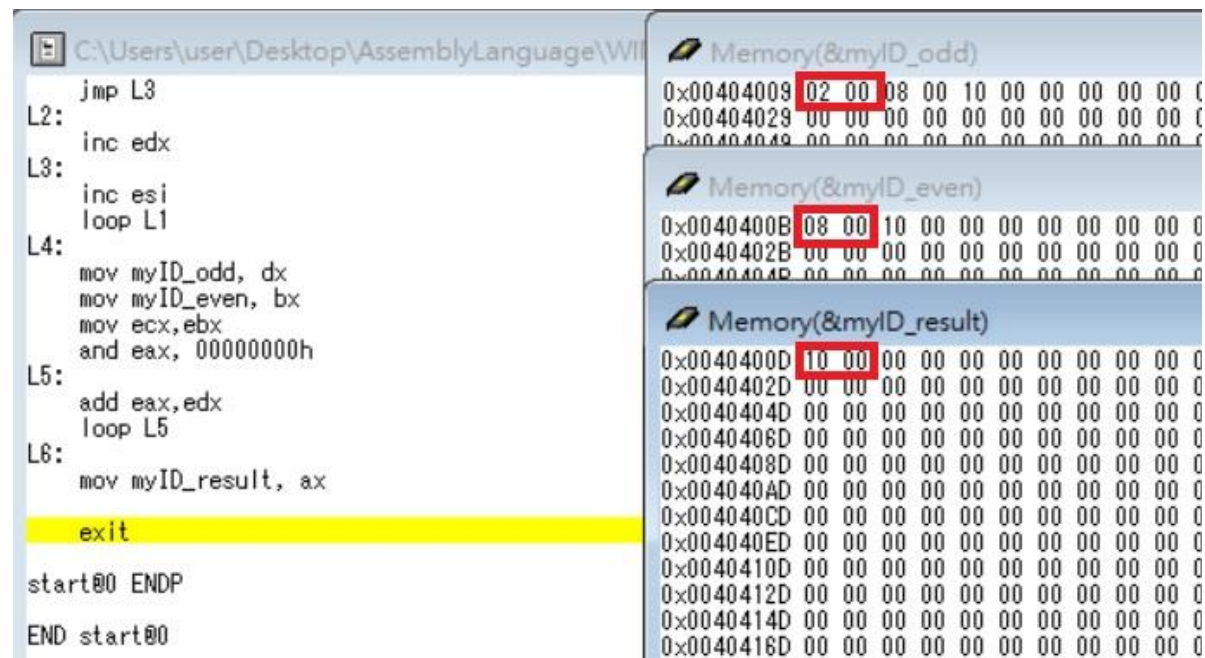
.data
    myID BYTE 1, 0, 2, 4, 0, 3, 0, 2, 0 ;其中一個人的學號
    size_ID = ($-myID) ;size_ID表示myID的長度
    myID_odd WORD ? ;myID中奇數數字的個數
    myID_even WORD ? ;myID中偶數數字的總和
    myID_result WORD ? ;myID_odd跟myID_even相乘的結果

.code
start@0 PROC
    mov ecx, size_ID
    xor esi, esi
    xor ebx, ebx
    xor edx, edx
L1:
    movzx eax, myID[esi]
    test eax, 1
    jnz L2
    add ebx, eax
    jmp L3
L2:
    inc edx
L3:
    inc esi
    loop L1
L4:
    mov myID_odd, dx
    mov myID_even, bx
    mov ecx, ebx
    and eax, 00000000h
L5:
    add eax, edx
    loop L5
L6:
    mov myID_result, ax

    exit
```

First, we moved ID number to eax and then set 102403020 as myID.

Then we moved myID to eax, a number at once and tested the value of eax was odd or even. If eax was odd, we added 1 to edx for counting times of odd number. ("inc edx") If eax is even, we added eax to ebx for counting the sum of even numbers. ("add ebx, eax") Finally, we used a loop to implement multiplication and put the result in myID_result.(L5)



The screenshot displays a debugger window with assembly code on the left and memory addresses on the right. The assembly code includes labels L2 through L6, with L5 highlighted in yellow. The memory addresses on the right show the values of variables myID_odd, myID_even, and myID_result, with specific values highlighted in red boxes.

```

C:\Users\user\Desktop\AssemblyLanguage\W...
jmp L3
L2:  inc edx
L3:  inc esi
    loop L1
L4:  mov myID_odd, dx
    mov myID_even, bx
    mov ecx, ebx
    and eax, 00000000h
L5:  add eax, edx
    loop L5
L6:  mov myID_result, ax
    exit
start@0 ENDP
END start@0

```

Memory(&myID_odd)

| | | | | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0x00404009 | 02 00 | 08 00 | 10 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x00404029 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x00404049 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |

Memory(&myID_even)

| | | | | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0x0040400B | 08 00 | 10 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040402B | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040404B | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |

Memory(&myID_result)

| | | | | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0x0040400D | 10 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040402D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040404D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040406D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040408D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x004040AD | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x004040CD | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x004040ED | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040410D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040412D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040414D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |
| 0x0040416D | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 | 00 00 |

After the program finished, we could see 2h in myID_odd (means 2 odd number in ID), 8h in myID_even(means the sum of even number is 8), 10h in myID_result.(means $2 \times 8 = 16$)

Review:

At first, we thought this exercise would be difficult because it looked like we needed to take care of many things. But after we discussed this question, we found it was really easy if we applicate the knowledge we just learned in today class. We just needed 3 loop to finish add 1 or add sum (odd or even) and 1 loop to implement multiplication. But we also made a mistake in this exercise: We put "mov ecx" and "size_ID" in the loop, which made our loop was not able to break. After this assignment, we learned how to control a procedure like an "if else" instruction.