111_HW2-Procedure

資電院學士班二年級

110504517 李睿穎

Full Code

```
include Irvine32.inc

.data
BitStrs BYTE 8 dup(?)
ChStrs BYTE "########"
       BYTE "      ##"
       BYTE "     ## "
       BYTE "   ##   "
       BYTE "   ##   "
       BYTE "   ##   "
       BYTE "   ##   "
       BYTE "   ##   "

.code
change PROC
        SHL BitStrs[edi], 1
        CMP ChStrs[esi], "#"
        JNE notEqual
        INC BitStrs[edi]
        notEqual:

        INC esi
        RET
change ENDP

main PROC
    MOV ecx, 8
    MOV esi, 0 ; use esi as ChStrs's  index
    MOV edi, 0 ; use edi as BitStrs's index

ConvertRow:
    Push ecx
    MOV ecx, 8
    ConvertColumn:
        CALL change
        LOOP ConvertColumn
    MOVZX eax, BitStrs[edi]

    MOV ebx, 1
    CALL WriteBinB
    CALL Crlf

    INC edi
    Pop ecx
    LOOP ConvertRow

    exit
main ENDP
END main
```

Procedure “change”

```
14  .code
15  change PROC
16          SHL BitStrs[edi], 1
17          CMP ChStrs[esi], "#"
18          JNE notEqual
19          INC BitStrs[edi]
20          notEqual:
21
22          INC esi
23          RET
24  change ENDP
```

Shift left BitStrs for next input value (store old number)
Compare target (ChStar[esi]) and “#”
If target != “#” , jump to notEqual and do nothing (0)
Increase BitStrs as input (1) value

Increase esi to next index

main

```
26  main PROC
27      MOV ecx, 8
28      MOV esi, 0 ; use esi as ChStrs's  index
29      MOV edi, 0 ; use edi as BitStrs's index
30
31  ConvertRow:
32      Push ecx
33      MOV ecx, 8
34      ConvertColumn:
35          CALL change
36          LOOP ConvertColumn
37      MOVZX eax, BitStrs[edi]
38
39      MOV ebx, 1
40      CALL WriteBinB
41      CALL Crlf
42
43      INC edi
44      Pop ecx
45      LOOP ConvertRow
46
47      exit
48  main ENDP
49  END main
```

Init ecx as loop count for Row

Store ecx value by Push into stack (for Row)
Init ecx as loop count for Column

Call “change” procedure
Loop until finish this column (8 times)

Set ebx as WriteBinB’s parameter (means write 1Byte)
Call method from Irvine32.inc
Next line

Increase edi for next Row (next element in BitStrs)
Get ecx value by Pop from stack (for Row)
Loop until finish this row (8 times)

END

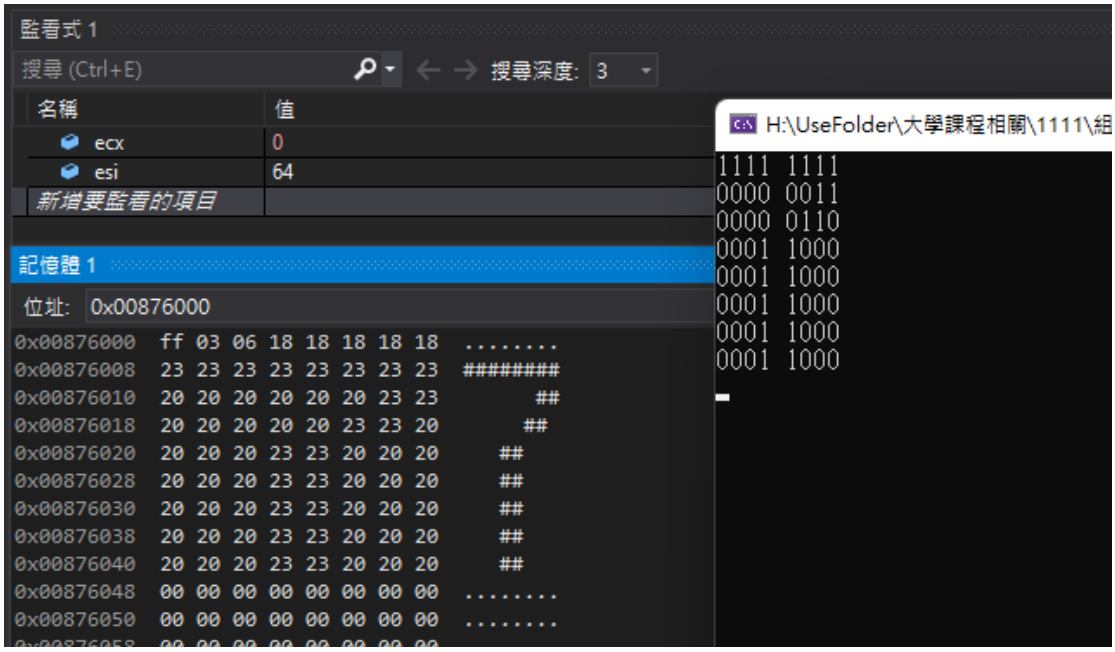
WriteBinB’s document

WriteBinB PROC (Not covered in the 4th edition)

Writes an unsigned 8, 16, or 32-bit number to standard output in ASCII binary format.
EBX must contain the TYPE of the number to write (1 for BYTE, 2 for WORD, or 4 for DWORD)
The binary bits are displayed in groups of 4 for easy reading.

```
Call args:  AL, AX, or EAX = the number to write.
            EBX = 1 to write AL as 8 binary digits,
            EBX = 2 to write AX as 16 binary digits,
            EBX = 4 to write EAX as 32 binary digits.

Return arg: None

Example:

      .data
      bValue BYTE 'A'

      .code
      mov  eax,bValue
      mov  ebx,TYPE bValue
      call WriteBinB

Output:  0100 0001

Notes: To write a DWORD, the WriteBin procedure may also be used.
       To write in hexadecimal, use the WriteHexB procedure.
       The mShow macro causes a call to this procedure.
```

Result Screenshot



According to this work, we can practice the use of Shift or Rotate, and nested loop.

But I think this is not a good question for practice Procedure, because in this question we don't really need to use Procedure or even don't use will let code better to read.

I think convert more number can be a solution, like make a procedure to convert a number, than input last 4 digit of student number.